Juridiska institutionen
Handelshögskolan vid Göteborgs universitet
HRO800 Examensarbete, 30 hp
Juristprogrammet
Vårterminen 2019

Reverse Engineering of Software

- An analysis of the possibility to contractually restrict reverse engineering of trade secrets in software

Rikard Kåresen

GÖTEBORGS UNIVERSITET
HANDELSHÖGSKOLAN

Handledare: Jens Andreasson

Summary

This thesis addresses the possibility to contractually restrict reverse engineering of trade secrets in software in the light of a systematic incoherence between two EU directives, the Software Directive and Trade Secret Directive. Mentioned directives are namely incoherent regarding the permissibility of contractual restrictions of reverse engineering which causes uncertainty about license provisions' legitimacy.

The purpose of this thesis is to address if one should be able to contractually restrict reverse engineering of trade secrets in software, and if so – to what extent. The thesis will also suggest contractual provisions for restricting reverse engineering which can be deemed legitimate based on the performed analysis of this thesis. The analysis and conclusions rely on a legal assessment of the EU directives, preparatory works, case law, as well as legal literature.

Furthermore, mentioned incoherence reflects a conflict of incentives for innovation which is strongly reflected in the software industry. The analysis is therefore nuanced by a literature review of economic and technological aspects of software and reverse engineering with the purpose of enhancing the relevance of the thesis in the software industry. The assessment is also done in relation to a fictitious case in order to make the analysis more legible and concrete.

Dictionary

CJEU	Court of Justice of the European Union.
Commission	European Commission.
Decompilation	A method of reverse engineering consisting of translating object code of software into source code.[1]
Reverse engineering	The art and practice of disassembling a product in order to find out how it functions and is constructed.[2]
SD	Software Directive, Council Directive 91/250/EEC of 14 May 1991 on the legal protection of computer programs. Directive 2009/24/EC of the European Parliament and of the Council of 23 April 2009 on the legal protection of computer programs (Codified version).
TRIPS	The Agreement on Trade-Related Aspects of Intellectual Property Rights.
TSA	Swedish Trade Secret Act (2018:558).
TSD	Trade Secret Directive, EU Directive 2016/943 of the European Parliament and of the Council of 8 June 2016 on the protection of undisclosed know-how and business information (trade secrets) against their unlawful acquisition, use and disclosure.

[1] Litman, J. (1992). *Copyright and Information Policy.* Law and Contemporary Problems, Vol. 55: No 2. 185-209, p. 197.
[2] Maunsbach, U. Wennersten, U. (2018). *Grundläggande immaterialrätt.* Malmö: Gleerups Utbildning AB, p. 109.

Index

1 Introduction

1.1 Background

Reverse engineering – the art and practice of disassembling a product in order to find out how it functions and is constructed, enables one to extract underlying know-how of the analyzed product.[3] The practice has been established as a legitimate way of obtaining information in industries ranging from the traditional manufacturing industries to the software industry.[4] However, the practice is far from uncontroversial and has been under a lot of debate. The debate has been especially prominent in the software industry, wherein opposers of the practice of reverse engineering says it undermines incentives for innovation, whereas the promoters claim that reverse engineering is necessary for sustaining fair competition in the software industry.[5]

The debate surrounding reverse engineering in the software industry was settled in the EU through the Software Directive, *Council Directive 91/250/EEC of 14 May 1991 on the legal protection of computer programs*, (SD), which established reverse engineering as a legitimate practice back in 1991.[6]

However, with the EU's recent Trade Secret Directive, *EU Directive 2016/943 of the European Parliament and of the Council of 8 June 2016 on the protection of undisclosed know-how and business information (trade secrets) against their unlawful acquisition, use and disclosure*, (TSD), the question of reverse engineering's legitimacy has been brought back to life. The TSD and SD are namely incoherent regarding the possibility to contractually restrict reverse engineering of trade secrets in software, which may affect how license agreements of software can be designed.[7]

The legitimacy of contractual provisions restricting reverse engineering of software is not an uncomplicated matter as the problem relates to different industry interests and several legal areas,

[3] Samuelsson, P. Scotchmer, S. (2002). *The Law and Economics of Reverse Engineering*. The Yale Law Journal, volume 111:1575, 1575–1663, p. 1607, and Maunsbach, U. Wennersten, U. (2018). *Grundläggande immaterialrätt*. Malmö: Gleerups Utbildning AB, p. 109.
[4] Samuelsson, P. Scotchmer, S. (2002). *The Law and Economics of Reverse Engineering*. The Yale Law Journal, volume 111:1575, 1575–1663, pp. 1577 and 1579.
[5] Palmer, A. K., Vinje, T. C. (1992). *The EC Directive on the legal protection of computer software: New law governing software development.* Duke Journal of comparative & international law, volume 2:65, 65-87, p. 71.
[6] Ibid, p. 78.
[7] See Shemtov, N. (2017). *Beyond the Code Protection of Non-Textual Features of Software.* Oxford: Oxford University Press, p. 229.

including contracts, copyright and trade secrets, which all have their own take on the matter. Nevertheless, this thesis will take on mentioned challenge.

1.2 Problem

The problem statement of this thesis, simply put, is that it is unclear if one can contractually restrict reverse engineering of software for protecting underlying know-how as trade secrets. Mentioned problem is caused by an incoherence between the TSD and SD which includes articles that promote two different standings on the matter, namely that reverse engineering can be, and that it cannot be contractually restricted.

The basis for contractually restricting reverse engineering stems from the principle of freedom of contract which sets out the right to freely decide a contract's content.[8] The principle does however defer to intervening and compulsory regulations such as the SD which sets out rights to reverse engineer and decompile software in article 5(3) and 6 of the SD, (decompilation of software is a method of reverse engineering in which object code is translated into source code, see more about decompilation in section 2.3.4). Contractual restrictions of the mentioned rights of reverse engineering is furthermore said to be deemed void and null according to article 8(2) of the SD which imposes that the freedom of contract principle is limited by the SD. However, if trade secrets are put into the equation, it becomes unclear if the SD's articles of reverse engineering are relevant, since article 8(1) of the SD states that the directive shall not affect regulations relating to trade secrets.

The TSD is namely legitimizing contractual restrictions of reverse engineering in recital 16 and article 3(1)(b) and (d), 4(2), and 4(3)(b) and (c) of the TSD. Mentioned recital does also stipulate: "The freedom to enter into such contractual arrangements can, however, be limited by law." The basis of the TSD is nevertheless that reverse engineering is a legitimate practice for obtaining information according to articles 3(1)(a) and (b) in relation to the 16th recital of the TSD.

The TSD's standpoint on reverse engineering can thus be summarized as follows:[9]

- Main rule: Reverse engineering is allowed
- Exception I: Reverse engineering can be contractually restricted

[8] About freedom of contract see e.g. Adlercreutz, A., Gorton, L., & Lindell-Frantz, E. (2016). *Avtalsrätt 1.* Lund: Juristförlaget, p. 34.
[9] See Schröder, V. (2018). *Transformation from a Patchwork Quilt to a Unified Fabric: Discussion on a Few Particular Aspects of the New Finnish Trade Secrets Act and the EU Trade Secrets Directive*. Nordiskt Immateriellt Rättsskydd, NIR 4 / 2018, 500-515, and Schröder, V. (2017, January). *Decompiling the rules on trade secrets, software and reverse engineering.* Downloaded 2019-02-03, from https://trustinip.com/decompiling-the-rules-on-trade-secrets (website no longer available).

- Exception II: The right to conclude contracts that restrict the practice of reverse engineering can be restricted through law

The icing on the cake is that recital 39 of the TSD states that the directive shall not affect other intellectual property rights regulations. Hence, it is apparent that there is a systematic incoherence between the two directives, since both directives promote different standings on the possibility to contractually restrict reverse engineering, and each directive is also stating that it shall not affect the other. License agreements restricting the practices of reverse engineering and decompilation might thus be legitimate, or be deemed void and null – it is currently unclear.

Moreover, with recent papers presenting studies of how one may reverse engineer software to obtain know-how such as machine learning algorithms by merely using the software, it is highly relevant to address the presented problem.[10] This applies not the least for the software industry in which companies often are dependent on abstract information such as AI/machine learning algorithms which might be obtained through reverse engineering.[11]

1.3 Purpose

The purpose of this thesis is to address if one should be able to contractually restrict reverse engineering and decompilation of trade secrets in software, and if so – to what extent. The analysis will stem from the incoherence between the two above mentioned directives, the SD and TSD, and interests and incentives within the software industry. Lastly, I will suggest contractual provisions for restricting reverse engineering and decompilation that can be deemed legitimate based on the performed analysis of this thesis.

1.4 Research Question

Should reverse engineering and decompilation of trade secrets in software be possible to contractually restrict, and if so, to what extent?

a) How do the TSD and SD interrelate with software?
b) Should one rely on the TSD or SD regarding the permissibility of contractual restrictions of reverse engineering and decompilation?

---

[10] See Augustin, M., Fritz, M., Oh, S. J., & Schiele, B. (2018). *Towards Reverse-Engineering Black-Box Neural Networks*. ICLR 2018 Conference paper, and Juels, A., Reiter, M. K., Ristenpart, T., Tramér, F., & Zhang, F. (2016). *Stealing Machine Learning Models via Prediction APIs.* Available through USENIX, Open Access Media.

[11] See PWC. (2019). *Why AI and robotics will define New Health*. Downloaded 2019-04-23 from https://www.pwc.com/gx/en/industries/healthcare/publications/ai-robotics-new-health.html

c) Based on a) and b), to what extent should reverse engineering and decompilation be possible to contractually restrict?

1.5 Delimitations

The research question will focus on reverse engineering and decompilation in license relationships. Hence, I will not address the practices in relation to transfers such as purchases, proceeds, or gifts.

Furthermore, even though this thesis will relate to software interface information, I will not try to conclude whether such information is subject to copyright protection or not.

The thesis will not cover aspects of competition law of reverse engineering and decompilation despite its relevance for the subject matter. I have chosen to exclude the aspect in order to enable an in-depth analysis of already set out research question under the limited time frame that I have for writing this thesis.

The thesis will focus on the regulations of the EU, and not include a comparative analysis of other jurisdictions because of the limited time I have for writing this thesis. I am aware that the relevance of this thesis would increase if it also related to other jurisdictions such as the US since software licenses often are distributed on a global level, requiring license agreements to comply with several jurisdictions. However, a comparative analysis of the EU and e.g. US regulations regarding reverse engineering and decompilation might be a good subject for another thesis.

Although technical protection measures are highly relevant in relation to reverse engineering, I have chosen to exclude the matter and dedicate the focus of this thesis on legal regulations and contractual matters. Furthermore, the legality of "shrink-wrap" licenses will not be assessed in this thesis.[12]

1.6 Method

I have chosen to address the research question (RQ) that stems from the incoherence between the SD and TSD, by reconstructing, interpreting and analyzing the three legal areas of contract, copyright and trade secrets, in relation to the subject matters of software and reverse engineering (including

[12] See Petrusson, U. (2004). *Intellectual Property & Entrepreneurship Creating Wealth in an Intellectual Value Chain.* Gothenburg: CIP, pp. 36-38 for an explanation of the license construction of “shrink-wrap”.

decompilation).[13] In the *Background* chapter, the three legal areas are addressed independently in order to determine how they generally apply and relate to the mentioned matters.[14] The incoherence between the SD and TSD is also specifically addressed by performing a literature review of legal literature which address the incoherence. The Background chapter is thus of a de lege lata[15] character, in which effective and current law is being addressed.

In the *Analysis* chapter, I address the three legal areas jointly in relation to the subject matters of software and reverse engineering in order to address how the mentioned legal areas specifically apply and relate to the matters.[16] The problem statement of this thesis is however implying that the legal situation is unclear and not yet solved, which makes the argumentation in the analysis of a de lege ferenda[17] character in RQ(b) and (c) – addressing how the problem *should* be solved.[18] The analysis concerning RQ(a) will however remain de lege lata. The analysis is also performed in relation to the performed literature review of the incoherence between the SD and TSD, in order to increase the analysis' relevance and reliability in the legal debate.[19]

Furthermore, since the purpose of this thesis is to address if one *should* have the possibility to contractually restrict reverse engineering and decompilation of trade secrets in software, I have also chosen to perform a literature review covering the subject matters of software and reverse engineering with a focus on technology and economic aspects. The reason for doing so is to enable a more industry relevant answer to whether reverse engineering *should* be possible to contractually restrict or not. The argumentation in this thesis is thus of a freer character compared to an analysis which strictly relates to legal sources, since arguments stemming outside of the legal sources are permitted in this thesis.[20]

Hence, the mentioned matters of software and reverse engineering with focus on technology and economic aspects, will be addressed in an introductory *Frame of Reference* chapter. The *Frame of Reference* chapter is concluded by introducing the reader to a fictitious practical example that will serve

[13] See Kleineman, J. (2018). Rättsdogmatisk metod, Nääv, M., & Zamboni, M. (Red.), *Juridisk metodlära*. Lund: Studentlitteratur AB, pp. 21 and 30.
[14] Ibid.
[15] De lege lata means the legal situation as it currently exists, see Nääv, M., & Zamboni, M. (2018). *Juridisk metodlära*. Lund: Studentlitteratur AB, p. 36.
[16] See Kleineman, J. (2018). Rättsdogmatisk metod, Nääv, M., & Zamboni, M. (Red.), *Juridisk metodlära*. Lund: Studentlitteratur AB, pp. 21 and 30.
[17] De lege ferenda means how the legal situation *should* be, e.g. how an unsolved legal problem should be solved, see Kleineman, J. (2018). Rättsdogmatisk metod, Nääv, M., & Zamboni, M. (Red.), *Juridisk metodlära*. Lund: Studentlitteratur AB, p. 36.
[18] See Kleineman, J. (2018). Rättsdogmatisk metod, Nääv, M., & Zamboni, M. (Red.), *Juridisk metodlära*. Lund: Studentlitteratur AB, p. 36, and Sandgren, C. (2018). *Rättsvetenskap för uppsatsförfattare ämne, material, metod och argumentation*. Stockholm: Norstedts Juridik AB. p. 49 wherein Sandgren elaborates on how de lege ferenda arguments can stem from a de lege lata assessment.
[19] See Sandgren, C. (2018). *Rättsvetenskap för uppsatsförfattare ämne, material, metod och argumentation.* Stockholm: Norstedts Juridik AB, pp. 36-37.
[20] See Kleineman, J. (2018). Rättsdogmatisk metod, Nääv, M., & Zamboni, M. (Red.), *Juridisk metodlära*. Lund: Studentlitteratur AB, p. 28.

as a way of contextualizing the RQ of this thesis. The industry relevance of the thesis is furthermore increased by having the *Analysis* of the thesis reconnecting to the mentioned practical example.

1.6.1 Frame of Reference

The *Frame of Reference* chapter starts off by elaborating on a software application's different components based on a performed literature review. Furthermore, the practice of reverse engineering is addressed by describing the purpose and meaning of the practice as well as related incentives in the software industry. The practice of reverse engineering is also elaborated on by describing two prominent methods of the practice, namely black-box reverse engineering and decompilation.

The literature review relies on legal literature which is useful and relevant since the descriptions of software and reverse engineering are not overly technologically detailed which helps describing the matters in this thesis with high legibility. Legal economic literature has also been used in order to address economic consequences and incentives relating to software and reverse engineering with the purpose of increasing the industry relevance of the thesis. The authority and reliability of the literature have been managed by relying on several publications published by prominent publishers and in scientific journals, which enables me to get a multifaceted understanding of the matters. The authority has also been managed by relating to publications with significant amount of citations and references.[21]

The literature review is also relying on non-legal literature, consisting of debated scientific reports, technological descriptions in dictionaries, scientific publications and industry reports, which are relevant for increasing the industry relevance of this thesis. This applies not the least for this thesis, since the legal literature consists of some dated publications of software and reverse engineering. I have relied on several publications which have been published by prominent publishers/journals enabling me to get a nuanced understanding of the matters of software and reverse engineering, similar to how I have managed the authority and reliability of the legal literature.[22]

1.6.2 Background and Analysis

The *Background* starts off by addressing the legal area of contracts with its entailing fundamental principle of freedom of contract in relation to software. The material for addressing contract law is legal literature, which to some extent is Swedish such as the book Avtalsrätt I by Axel Adlercreutz.[23] The legal literature which has been used is both acknowledged as being relevant and having significant

[21] See Sandgren, C. (2018). *Rättsvetenskap för uppsatsförfattare ämne, material, metod och argumentation*. Stockholm: Norstedts Juridik AB, pp. 36-37.
[22] Ibid.
[23] Adlercreutz, A., Gorton, L., & Lindell-Frantz, E. (2016). *Avtalsrätt 1.* Lund: Juristförlaget.

authority within the legal field of contract law.[24] However, one may question the relevance of using Swedish literature in this EU-based assessment, but the principles addressed are fundamental with high internal validity and apply on an international level including the EU. I have also relied on Swedish preparatory works in a limited extent, for underpinning an assumption in this thesis, which is done knowing that the mentioned material does not have legal bearing.

Furthermore, I address the legal area of contracts in relation to reverse engineering by using a prominent legal publication *Beyond the Code: Protection of Non-Textual Features of Software* by Dr. Noam Shemtov who is active at the Centre for Commercial Law Studies at Queen Mary University of London as a Senior Lecturer in Technology Law and Intellectual Property.[25] In mentioned publication, Shemtov has compiled several contractual provisions that restrict reverse engineering and decompilation. The contractual provisions come from prominent actors within the software industry and Shemtov's publication is published through Oxford University Press.[26] This indicates both high authority as well as reliability for the book and its addressed contractual provisions. Shemtov's publication is also relevant since Shemtov addresses the incoherence of the SD and TSD from an EU perspective with a logic description giving substantial information on how one may interpret the legal sources as well as incoherence.[27]

The *Background* is then continuing with addressing the two legal areas of copyright and trade secrets in relation to software and reverse engineering (including decompilation). I do also elaborate on the underlying purpose of having a right to reverse engineer in the SD and TSD, which is influenced by the teleological method used by the Court of Justice of the European Union (CJEU). Teleological interpretation is one of the interpretation models that the CJEU is known for using when there is an incoherence or ambiguity in the regulations. The method is characterized by heavily considering the purpose of the regulations when interpreting mentioned incoherencies or ambiguities. It is suggested that the teological interpretation fulfills the following purposes:[28]

- Promoting the purpose of a provision
- Preventing unreasonable consequences of a more literal interpretation

[24] See Sandgren, C. (2018). *Rättsvetenskap för uppsatsförfattare ämne, material, metod och argumentation.* Stockholm: Norstedts Juridik AB, pp. 36-37.
[25] Oxford University Press. (2019). *Beyond the Code Protection of Non-Textual Features of Software.* Downloaded 2019-04-23 from https://global.oup.com/academic/product/beyond-the-code-9780198716792?cc=se&lang=en&#
[26] Shemtov, N. (2017). *Beyond the Code Protection of Non-Textual Features of Software.* Oxford: Oxford University Press.
[27] See Sandgren, C. (2018). *Rättsvetenskap för uppsatsförfattare ämne, material, metod och argumentation.* Stockholm: Norstedts Juridik AB, pp. 36-37.
[28] See Eriksson, I. O., & Hettne, J. (2011). *EU-rättslig metod Teori och genomslag i svensk rättstillämpning.* Stockholm: Norstedt Juridik AB, p. 168.

- Filling out gaps in the EU regulations

The promotion of a provision's purpose can be considered to be the most common purpose for using teological interpretation, which imply that it is relevant to rely on such an interpretation in the analysis when addressing the problem statement which is a result of an incoherence between two EU directives.[29] The material for addressing the legal areas of copyright and trade secrets, are primarily the directives with their articles and recitals. I have thus also been influenced by the EU-methodology for valuing legal sources and choosing relevant material.[30] I am aware that the recitals do not have a normative value in themselves, which is why they are related to for interpreting articles and their underlying purposes.[31] I have also used preparatory works for interpreting the directives, since preparatory works have a standing within the EU-methodology for valuing legal sources.[32] Legal literature was chosen based on its expressed reasoning and logic, which is similar to how EU relates to legal literature.[33] Furthermore, I have relied upon legal publications which describe the procedural enactment process of the SD in order to outline the meaning and purpose of the directive's articles. Mentioned publications refer heavily to preparatory works from EU institutions and are published on prominent legal journals with substantial amounts of references, thus, indicating significant relevance, authority and reliability.[34]

A case from the CJEU, namely a preliminary ruling, has also been used for interpreting articles of the SD and their entailing purposes.[35] Mentioned case was given in the Grand Chamber, thus indicating a significant authority within EU with entailing relevance and reliability.[36] The ruling is binding for the court referring the preliminary ruling, however the CJEU does not consider itself to be bound by case-law, which should be taken into consideration when addressing how the legal situation might evolve over time.[37] I have also relied on the General Advocate's opinion in relation to the addressed case law. The General Advocate's opinion does not have the same legal authority as case law since the opinion is

[29] See Ibid, p. 169.
[30] See Ibid, p. 40.
[31] See European Parliament Council Commission, Interinstitutional Agreement of 22 December 1998 on common guidelines for the quality of drafting of Community legislation, General Principles 10.
[32] See Eriksson, I. O., & Hettne, J. (2011). *EU-rättslig metod Teori och genomslag i svensk rättstillämpning*. Stockholm: Norstedt Juridik AB, pp. 113-116, in which it is stated that preparatory works has a standing as a legal source, although not of the same authority as in the Swedish legal system.
[33] Ibid, p. 120.
[34] Referred to articles are: Palmer, A. K., Vinje, T. C. (1992). *The EC Directive on the legal protection of computer software: New law governing software development.* Duke Journal of comparative & international law, volume 2:65, 65-87, and Dreier, T. (1991). *The Council Directive of 14 May 1991 on the Legal Protection of Computer Programs.* European Intellectual Property Review. 13(9) 319-330.
[35] Case C-406/10, *SAS Institute Inc. v. World Programming Ltd.*
[36] Eriksson, I. O., & Hettne, J. (2011). *EU-rättslig metod Teori och genomslag i svensk rättstillämpning*. Stockholm: Norstedt Juridik AB, p. 58.
[37] Ibid, p. 51.

not a judgement. However, if there are referrals in the judgement from the CJEU to the opinion of the General Advocate, both the authority as well as relevance increase.[38]

In relation to trade secrets, I have also addressed the Agreement on Trade-Related Aspects of Intellectual Property Rights (TRIPS) which is an international agreement for the members of the World Trade Organization (WTO) which the EU and its member states are a part of. The agreement is binding and serves as a minimum protection for intellectual property for the members, which makes the agreement relevant to address when interpreting EU directives.[39]

Lastly, after I have addressed the three legal areas, I address the incoherence of the SD and TSD based on the expressed opinions in legal literature. The legal literature will be Shemtov's mentioned publication, a publication by Vilhelm Schröder who is an Attorney-at-Law, LL.M. IP and LL.D. Candidate, and a publication by Thomas Dreier who is a Professor of Law at the University of Karlsruhe/Karlsruhe Institute of Technology.[40]

Schröder's publication was published at Nordiskt Immateriellt Rättsskydd (NIR) which is a prominent legal review in the Nordics indicating both significant reliability as well as authority to the opinions Schröder expressed. It was the publication which influenced me to write this thesis in the first place, hence indicating a substantial relevance for this thesis.[41] Dreier published a comment on the SD in the European Intellectual Property Review which achieved a significant amount of citations. Considering the foregoing it is apparent that Dreier's publication has both reliability, authority and relevance for this thesis.

Subsequently I have related to the following hierarchy of the sources addressed:

I. International treaties

II. EU Directives

III. CJEU case law

IV. EU Preparatory works

---

[38] Ibid, pp. 116-117.
[39] Levin. M. (2017). *Lärobok i immaterialrätt.* Stockholm: Wolters Kluwer Sverige AB, pp. 49-50.
[40] Schröder, V. (2018). *Transformation from a Patchwork Quilt to a Unified Fabric: Discussion on a Few Particular Aspects of the New Finnish Trade Secrets Act and the EU Trade Secrets Directive.* Nordiskt Immateriellt Rättsskydd, NIR 4 / 2018, 500-515.
[41] Ibid.

V. Legal literature

As for the analysis, I start off with addressing RQ (a), namely how the SD and TSD interrelate with software. I then move over to RQ (b), by addressing which one of the two directives one should relate to when addressing the permissibility of contractual restrictions of reverse engineering and decompilation. RQ(c) is then addressed – to what extent should one be able contractually restrict reverse engineering and decompilation? The analysis will consistently reconnect to the practical example of this thesis by also addressing to what extent the company of mentioned example should be possible to restrict reverse engineering and decompilation in practice. The analysis under RQ(c) is furthermore concretized by suggesting contractual clauses which promotes the interests of not having trade secrets obtained through reverse engineering or decompilation.

1.7 Disposition

The disposition of this thesis is as follows, Introduction (chapter 1), Frame of Reference (chapter 2), Background (chapter 3), Analysis (chapter 4), and Sources (chapter 5).

2 Frame of Reference

2.1 Introduction

The purpose of this chapter is to give the reader a conceptual understanding of a generic software application. The purpose is furthermore to address reverse engineering in general, industry interests and incentives in the software industry, and to elaborate on technical aspects of reverse engineering by describing two prominent methods of the practice. Lastly, this chapter will have a fictitious yet practical example in order to contextualize, concretize, as well as limit the RQ of this thesis.

2.2 Software Components

Software is a term for the intangible components of a computer system and can be divided and assessed in many ways, but I have chosen to use the definitions that are embraced in the legal industry in order to facilitate the later legal analysis.[42] The main components of a software application can be said to be source code, object code, software interface, and abstract information such as ideas and principles.[43] The definitions are not mutually exclusive and are overlapping to some extent.

Source code consists of commands in human-readable text that has been written in programming languages, e.g. C++ or Java, which outline how a software should solve problems.[44] A common example of how software and source code function is the “Hello World! Software”, which was first introduced in the book of *C Programming Language* by Brian Kernighan and Dennis Ritchie.[45] You can write the "Hello World!" software in JavaScript language by writing a line in source code which enables the computer to generate the line “Hello World!”, see below:

> “document.write("Hello World!");“[46]

[42] Butterfield, A., & Ngondi, G. E. (2016). *A dictionary of Computer Science.* Oxford: Oxford University Press, Software.
[43] See Dreier, T. (1991). *The Council Directive of 14 May 1991 on the Legal Protection of Computer Programs.* European Intellectual Property Review. 13(9) 319-330, p. 6 in which Dreier relates to mentioned components in his assessment of the Software Directive.
[44] Shemtov, N. (2017). *Beyond the Code: Protection of Non-textual features of Software.* Oxford University Press, p. 73, and Johnson-Laird, A. (1994). *Software Reverse Engineering in the Real World.* Dayton. L. Rev. (19) 843-902, pp. 856 and 858-859.
[45] Kernighan, B. E., & Ritchie, D. M. (1988). *The C Programming Language.* New Jersey: Precentice Hall.
[46] W3Schools. (2019). *HTML DOM write() Method*. Downloaded 2019-04-24 from https://www.w3schools.com/jsref/met_doc_write.asp

The object code of a software is the translation of source code, such as the one exemplified above, into machine code in binary writing (ones and zeros) which only the computer can understand and execute in greater volumes.[47]

Software interface can be described as the part of a software application which has the function of interconnecting one part of a software to another, see recital 10 of the SD. For instance, if one would like to run a software application on the operating system of Windows, one would need to have information about Window's software interface in order to make the software compatible with Windows. Hence, having information about the software interface is essential for being able to develop a compatible/interoperable software.[48] It serves to be mentioned that software interface is a part of the software, existing both as source code and object code.

Ideas and principles on the other hand are the abstract information of a software, such as algorithms, design structure etc., which are embedded in the software's source and object code.[49] Andrew Johnson-Laird who is an English-American computer scientist with experience from testifying about software in Federal and District Courts in the US, describes the relation between the different components of a software with an example of steak sauce. The actual steak sauce that you have on your plate can be compared to the object code, wherein the source code is equivalent to the steak sauce recipe in written format. The recipe in itself as the chef remembers it in his mind, is the abstract information – ideas and principles.[50] However, there is often an interest in protecting the recipe in its abstract format, since that is often what contains value as know-how and not the textual expression/recipe itself.

The components of a software application can thus be summarized in the following illustration.

[47] Shemtov, N. (2017). *Beyond the Code: Protection of Non-textual features of Software.* Oxford University Press, p. 73, and Johnson-Laird, A. (1994). *Software Reverse Engineering in the Real World*. Dayton. L. Rev. (19) 843-902, pp. 856 and 858-859.
[48] Dreier, T. (1991). *The Council Directive of 14 May 1991 on the Legal Protection of Computer Programs.* European Intellectual Property Review. 13(9) 319-330, p. 6.
[49] See Johnson-Laird, A. (1994). *Software Reverse Engineering in the Real World*. Dayton. L. Rev. (19) 843-902, p. 856, in which he addresses abstract information of software.
[50] Johnson-Laird, A. (1994). *Software Reverse Engineering in the Real World.* Dayton. L. Rev. (19) 843-902, p. 851, and Johnson-Laird Inc. (2017). *About Andy*. Downloaded 2019-04-29 from https://www.jli.com/about.html

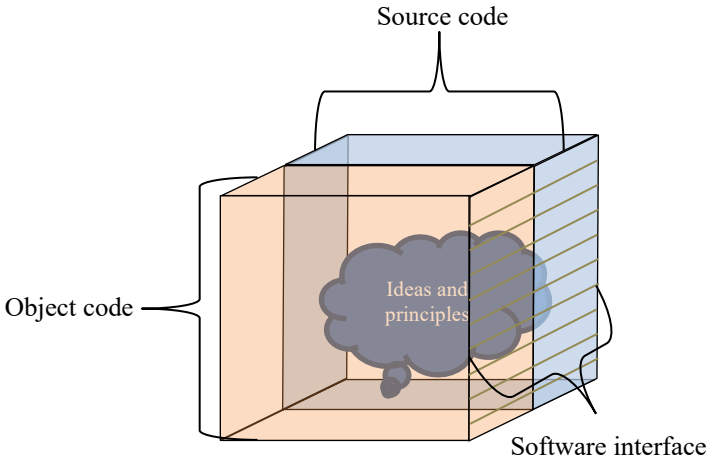

Figure 1: Illustration of a software application with its component, drafted by me.

2.3 Reverse Engineering

2.3.1 Introduction

Reverse engineering of software consists of backtracking how a software application works or how it was constructed.[51] The definition is wide and includes different methods of reverse engineering, including black-box reverse engineering and decompilation which I will describe in the two next coming sections.[52] According to Johnson-Laird, the reasons for performing reverse engineering can either be to figure out how the software application operates, or why it does not operate as intended.[53]

Shemtov has suggested that the reasons for wanting to understand how a software application operates are the following; (i) creation of a competing software, (ii) creation of a software that is interoperable/compatible with the analyzed software, (iii) to perform analysis for research purposes, and (iv) to perform security review.[54] Pamela Samuelsson a Professor of Law at University of California, Berkeley, and Suzanne Scotchmer Professor of Law, Economics and Public Policy at University of California, Berkeley, published the paper *The Law and Economics of Reverse Engineering,* wherein they claim that the prominent reason for performing reverse engineering in the software industry is reason (ii) creation of a software that is interoperable/compatible with the analyzed software. The

[51] Butterfield, A., & Ngondi, G. E. (2016). *A dictionary of Computer Science*. Oxford: Oxford University Press, Reverse engineering.
[52] See Shemtov, N. (2017). *Beyond the Code Protection of Non-Textual Features of Software.* Oxford: Oxford University Press, pp. 71-73.
[53] Johnson-Laird, A. (1994). *Software Reverse Engineering in the Real World*. Dayton. L. Rev. (19) 843-902, p. 846.
[54] Shemtov, N. (2017). *Beyond the Code Protection of Non-Textual Features of Software.* Oxford: Oxford University Press, p. 75.

prominence of reason (ii) is a result from the industry dynamic, since software generally needs to operate in an ecosystem of platforms, such as Windows, and other application running on the same platform.[55]

However, actors are also reverse engineering software applications with the purpose of developing competing products, implying that reason (i) also is relevant in the industry.[56] Moreover, the practical example in section 2.4 of this thesis focus on reverse engineering in relation to reason (i) wherein a competing product might be developed through reverse engineering.

2.3.2 Reverse Engineering in the Software Industry

Reverse engineering in the software industry is highly debated wherein opposers of the practice mean that reverse engineering undermine control of software causing a decrease of the incentive to innovate since innovators are not able to recoup their R&D costs, whereas promoters consider the practice to enable fair competition.[57] Hence, the question of reverse engineering's legitimacy in the software industry reflects a conflict between free competition, and control/exclusivity given by intellectual property in the software industry.[58]

The Royal Swedish Academy of Engineering Sciences (IVA) has elaborated on mentioned relationship between intellectual property rights and free competition which was concluded in an industry report. IVA suggested that intellectual property rights such as copyright is motivated by enabling incentives to innovate and to facilitate commercialization which also benefits consumers. Intellectual property rights are further said to be motivated by knowledge and technology distribution, and to facilitate creation of contracts, licenses and transactions. The drawbacks of intellectual property rights were suggested to be unnecessary double work within R&D, consumers paying higher prices for products and competitors being prevented from creating similar products.[59] One can of course question the premise that control is a driver for innovation, which has been done in other reports that conclude that intellectual property protection cannot be empirically proven to favor innovation and economic growth.[60]

---

[55] See Samuelsson, P. Scotchmer, S. (2002). *The Law and Economics of Reverse Engineering.* The Yale Law Journal, volume 111:1575, 1575-1663, p. 1615, Berkeley Economics University of California. (2019). *Suzanne Scotchmer*. Downloaded 2019-04-24 from https://www.econ.berkeley.edu/profile/suzanne-scotchmer and, Berkeley Law University of California. (2019). *Pamela Samuelsson.* Downloaded 2019-04-24 from https://www.law.berkeley.edu/our-faculty/faculty-profiles/pamela-samuelson/

[56] See Samuelsson, P. Scotchmer, S. (2002). *The Law and Economics of Reverse Engineering.* The Yale Law Journal, volume 111:1575, 1575-1663, p. 1580 in which the need for managing copying of information-based products is addressed.

[57] See Ibid, p. 1582, and Palmer, A. K., Vinje, T. C. (1992). *The EC Directive on the legal protection of computer software: New law governing software development.* Duke Journal of comparative & international law, volume 2:65, 65-87, p. 71.

[58] Maunsbach, U. Wennersten, U. (2018). *Grundläggande immaterialrätt.* Malmö: Gleeurups Utbildning AB, p. 109.

[59] Kungl. IngenjörsVetenskapsAkademien (IVA). (2011). *Immateriella Rättigheter och Ekonomiska Incitament – En kortfattad översikt av kunskapsläget.* Stockholm: Kungl. Ingenjörsvetenskapsakademien (IVA), p. 6.

[60] Levin. M. (2017). *Lärobok i immaterialrätt.* Stockholm: Wolters Kluwer Sverige AB, p. 35.

Pamela Samuelsson and Suzanne Scotchmer have also addressed the relationship between intellectual property rights and competition, but with focus on the legitimacy of reverse engineering, wherein they state that the economic effects of mentioned practice must be addressed in relation to the specific industry context.[61]

The software industry differs from the traditional manufacturing industries in several regards since software applications are knowledge-intensive products contrary to products from traditional manufacturing industries. Software applications are namely containing surrounding know-how to a greater extent compared to e.g. a product produced by a machine where the know-how remains in the manufacturing facility.[62] Thus, a software application can be said to be vulnerable against reverse engineering, or if we flip the argument – be suitable for reverse engineering, since a substantial amount of know-how is possible to extract from the practice.

Pamela Samuelsson and Suzanne Scotchmer have the opinion that the original innovator already has its innovation protected by the lead time on the market, since the practice of reverse engineering is both costly and time-consuming which imply that it takes a while before a competitor can introduce a competing product developed from reverse engineering on the market.[63] The lead time could even be compared to a short-lived intellectual property right, since one is given the monopoly on the market which one otherwise would have through e.g. copyright.[64] Such an argument shall of course be considered in relation to the specific premises of the software industry. Software is namely not as difficult and costly to produce compared to products requiring raw materials etc., which imply that the lead time in software may not be as extensive as in other industries.[65] Nevertheless, the time and costs for performing the practice of reverse engineering, is significant since the practice is deemed difficult and resource intensive, which may imply that there is a lead time for the innovator after all.[66]

Another aspect of reverse engineering is follow-on innovation. Follow-on innovation consists of building upon already existing innovation.[67] Reverse engineering can namely be said to facilitate follow-on innovation by promoting the interest of technology distribution in which researchers and commercial

---

[61] Samuelsson, P. Scotchmer, S. (2002). *The Law and Economics of Reverse Engineering*. The Yale Law Journal, volume 111:1575, 1575–1663, p. 1585.
[62] Ibid, p. 1579.
[63] Ibid, p. 1582.
[64] Ibid, p. 1586.
[65] See Davis, R., Kapor, M. D., Reichman, J. H., & Samuelsson. (1994). *A Manifesto concerning the Legal Protection of Computer Programs*. Columbia Law Review, Vol. 94:2308. 2308-2431, p. 2333.
[66] See Samuelsson, P. Scotchmer, S. (2002). *The Law and Economics of Reverse Engineering.* The Yale Law Journal, volume 111:1575, 1575–1663, p. 1613.
[67] Davis, R., Kapor, M. D., Reichman, J. H., & Samuelsson. (1994). *A Manifesto concerning the Legal Protection of Computer Programs.* Columbia Law Review, Vol. 94:2308. 2308-2431, p. 2409-2410.

actors can learn from existing technology through reverse engineering in order to create better and new innovations.[68] Hence, prohibition of reverse engineering could be said to cause wasteful costs on a societal level since companies would develop the same things without learning from each other resulting in negative impact on the market, which also reconnects to the incentive of knowledge and technology distribution presented by IVA.[69]

A way of wrapping up this section can be to refer to an interesting comment given in the IVA report; a way of addressing the opposing interests of competition and intellectual property rights is to weigh the benefits of society against the benefits of private actors. In more concrete terms; weigh the innovator's interests of having exclusive rights on the market, against the society's interests of having free competition, distribution of knowledge and new products developed.[70]

In the upcoming section, I will address two prominent methods of reverse engineering, namely black-box reverse engineering and decompilation which will be described in more detail below.

2.3.3 Black-Box Reverse Engineering

Black-box reverse engineering, a method of reverse engineering, consists of examining a software's behavior by testing different inputs (commands or queries) on a software application to see how the various inputs affect the results (output) of the software. By outlining the relationship between the input and output one can derive the underlying functions and ideas of the software.[71] Thus, it is apparent that black-box reverse engineering is not reliant on internal information of the software and the method is not requiring anything but using the software.[72]

In recent years, the method of black-box reverse engineering has been applied on software applications containing machine learning algorithms, in which the method has been proven to enable extraction of such algorithms.[73] The method presented in the article *Towards Reverse-Engineering Black-Box Neural Networks*, consisted of running a set of queries as input on a target software which generated corresponding output. The target software was analyzed by a mathematical model, called the "meta

---

[68] Ibid, p. 2393.
[69] Technology distribution is also addressed in Levin. M. (2017). *Lärobok i immaterialrätt.* Stockholm: Wolters Kluwer Sverige AB, p. 35, and Davis, R., Kapor, M. D., Reichman, J. H., & Samuelsson. (1994). *A Manifesto concerning the Legal Protection of Computer Programs.* Columbia Law Review, Vol. 94:2308. 2308-2431, p. 2393.
[70] Kungl. IngenjörsVetenskapsAkademien (IVA). (2011). *Immateriella Rättigheter och Ekonomiska Incitament En kortfattad översikt av kunskapsläget.* Stockholm: Kungl. Ingenjörsvetenskapsakademien (IVA), pp. 15-16.
[71] Butterfield, A., & Ngondi, G. E. (2016). *A dictionary of Computer Science.* Oxford: Oxford University Press, Black-box testing.
[72] Shemtov, N. (2017). *Beyond the Code Protection of Non-Textual Features of Software*. Oxford: Oxford University Press, p. 71.
[73] See Augustin, M., Fritz, M., Oh, S. J., & Schiele, B. (2018). *Towards Reverse-Engineering Black-Box Neural Networks.* ICLR 2018 Conference paper, and Juels, A., Reiter, M. K., Ristenpart, T., Tramér, F., & Zhang, F. (2016). *Stealing Machine Learning Models via Prediction APIs.* Available through USENIX, Open Access Media.

model", that parallelly ran the same input that had been run on the target software and mapped it to the output of the target software. The method enabled their metamodel to derive the used functions of the target software and "extract the machine learning algorithm".[74]

2.3.4 Decompilation

Decompilation is a method of reverse engineering, which consists of translating object code of a software application into source code.[75] The process is enabling the person who decompiles, to get access to source code, which embodies software interface information and abstract information such as ideas and principles, e.g. algorithms.[76] The method can be said to be a reversion of the development cycle of a software, where instead of writing the source code and then translating it into object code through the use of a 'compiler' (a translator program), you go the other way by translating the object code into source code.[77] One can divide the actions of decompilation into the stages of copying the software to the memory of the computer, translating the software's objective code into source code, and subsequently copying the translation.[78]

Thus, it is apparent that the method of decompilation is more intervening than the process of black-box reverse engineering since the latter only requires regular use of the software, and the former enables one to look into the software's components.

In the next section I will describe a fictitious example relating to software and reverse engineering including decompilation.

2.4 Example

Med-company Z is a three-year-old company and has a newly developed machine learning algorithm which enables a specific diagnostic method of X-ray pictures. The machine learning algorithm has been implemented in a software application which Med-company Z has used for generating diagnostic results of X-ray pictures. The company has yet only demonstrated the software application on exhibitions and conventions, and has not engaged in any commercial relationships. Med-company Z has received very positive feedback from their demonstrations and one specific company has approached Med-company

---

[74] Augustin, M., Fritz, M., Oh, S. J., & Schiele, B. (2018). *Towards Reverse-Engineering Black-Box Neural Networks.* ICLR 2018 Conference paper, pp. 2-3.
[75] Litman, J. (1992). *Copyright and Information Policy.* Law and Contemporary Problems, Vol. 55: No 2. 185-209, p. 197.
[76] Shemtov, N. (2017). *Beyond the Code Protection of Non-Textual Features of Software.* Oxford: Oxford University Press, p. 78, and Dreier, T. (1991). *The Council Directive of 14 May 1991 on the Legal Protection of Computer Programs.* European Intellectual Property Review. 13(9) 319-330, p. 6.
[77] Shemtov, N. (2017). *Beyond the Code Protection of Non-Textual Features of Software.* Oxford: Oxford University Press, p. 73.
[78] Litman, J. (1992). *Copyright and Information Policy.* Law and Contemporary Problems, Vol. 55: No 2. 185-209, p. 198.

Z with a suggestion to license the software from Med-company Z with the purpose of performing some early-stage trials. The offer is of course appealing to Med-company Z, but the company has some concerns about losing control over their machine learning algorithm.

Their concerns mainly come from the risk of having the algorithm reversed engineered through black-box reverse engineering or decompilation by a potential licensee which would enable the licensee to fully extract the algorithm and leave Med-company Z without their unique selling point. Med-company Z does not have the financial capital to file or uphold a patent for the software and is thus considering relying on copyright and trade secret protection in a potential licensing relationship.

Thus, it is present that in order to help Med-company Z, one needs to assess the RQ of this thesis, should reverse engineering and decompilation of trade secrets in software be possible to contractually restrict, and if so, to what extent?

Now that we have been introduced to the example, and received a conceptual understanding of the components of a software and the different methods of reverse engineering with entailing economic and technology aspects, we will move into the next chapter.

3 Background

3.1 Introduction

The purpose of this chapter is to independently address the three legal areas of contract, copyright and trade secret law in order to determine how they generally apply and relate to the matters of software and reverse engineering, including decompilation.[79] The incoherence between the SD and TSD is also specifically addressed based on a legal literature review.

3.2 Contract

3.2.1 Freedom of Contract

Contracts is a prominent mean for controlling and transacting objects such as the software application addressed in the example of the thesis, through e.g. a license agreement. The possibility to enter into contractual relationships and decide the contractual content follows from the principle of freedom of contract and pacta sunt servanda (agreements must be kept) which are two fundamental principles within contract law.[80] The principles are motivated by a turnover interest, in which the exchange of goods and utilities is being facilitated by contracts, and play a major part in the market economy.[81] Software license agreements are in particular subject to extensive freedom, since such license agreements does not adhere to a certain contractual type but are instead considered as their own contractual type *sui generis*.[82]

3.2.2 Contractual Provisions; Reverse Engineering

A way of hindering reverse engineering and decompilation is to include contractual provisions hindering such methods by relying on mentioned principle of freedom of contract. A common way is to include an anti-reverse engineering clause in the license agreement.[83] Another protective measure is to include a secrecy/non-disclosure provision to prevent certain use of the information that has been acquired.[84]

---

[79] See Kleineman, J. (2018). Rättsdogmatisk metod, Nääv, M., & Zamboni, M. (Red.), *Juridisk metodlära*. Lund: Studentlitteratur AB, pp. 21 and 30.
[80] Adlercreutz, A., Gorton, L., & Lindell-Frantz, E. (2016). *Avtalsrätt 1.* Lund: Juristförlaget, p. 28.
[81] See Ibid, p. 19 in which the societal functions of contracts are addressed.
[82] Bernitz, U., Pehrson, L., Rosén, J., & Sandgren, C. (2017). *Immaterialrätt och otillbörlig konkurrens*. Stockholm: Jure Förlag, p. 419.
[83] Shemtov, N. (2017). *Beyond the Code Protection of Non-Textual Features of Software.* Oxford: Oxford University Press, p. 11.
[84] See Fahlbeck, R. (2019). *Lagen om skydd för företagshemlighet En kommentar och rättsöversikter.* Stockholm: Norstedt Juridik AB, p. 122.

The following three contractual provisions are taken from an already existing compilation of anti-reverse engineering provisions, in which prominent software actors' provisions are presented.[85]

Intel has the following restriction in their Software License Agreement:[86]

> "You may not copy, modify, rent, sell, distribute or transfer any part of the Software except as provided in this Agreement, and you agree to prevent unauthorized copying of the Software.
>
> You may not reverse engineer, decompile, or disassemble the Software."

Apple has the following provision in their End-User-License-Agreement:[87]

> "You may not copy (except as expressly permitted by this license and the Usage Rules), decompile, reverse engineer, disassemble, attempt to derive the source code of, modify, or create derivative works of the Licensed Application, any updates, or any part thereof (except as and only to the extent any foregoing restriction is prohibited by applicable law or to the extent as may be permitted by the licensing terms governing use of any open sourced components included with the Licensed Application)."

Adobe has the following provision in their General Terms of Use agreement:[88]

> "You must not misuse the Services, Software, or content that we provide to you as part of the Services. For example, you must not:
>
> copy, modify, host, stream, sublicense, or resell the Services, Software, or content;"

What is apparent from these provisions is that they aim to prevent *acquisition* of valuable information rather than limiting a certain *use* of information obtained through reverse engineering. A non-disclosure/secrecy agreement on the other hand regulates the situation where information has been shared but it is agreed among the parties that the information is confidential and only to be *used* under certain premises in compliance with a set out purpose, see e.g. WIPO's NDA template.[89] The distinction

---

[85] Shemtov, N. (2017). *Beyond the Code Protection of Non-Textual Features of Software.* Oxford: Oxford University Press, p. 12.
[86] Ibid, referring to Intel Software License Agreement http:// www.intel.com/ design/ network/ drivers/ sla_ ec.htm?
[87] Shemtov, N. (2017). *Beyond the Code Protection of Non-Textual Features of Software.* Oxford: Oxford University Press, p. 12, referring to ITunes EULA http:// www.apple.com/ legal/ itunes/ appstore/ dev/ stdeula/
[88] Shemtov, N. (2017). *Beyond the Code Protection of Non-Textual Features of Software.* Oxford: Oxford University Press, p. 12, referring to Adobe General Terms of Use at http:// www.adobe.com/ legal/ terms.html
[89] See WIPO. (2019). *Disclosing Confidential Information*. Downloaded 2019-04-24 from https://www.wipo.int/sme/en/documents/disclosing_inf_fulltext.html#P53_4509

of restricting *acquisition* and *usage* will be considered in the later analysis in relation to the extent one may restrict reverse engineering, since the most extensive restriction would be to both restrict *acquisition* as well as *usage*.

A way of wrapping up the contract law section is to reconnect to the problem statement of this thesis, by stating that it is the above-mentioned contractual provisions that may be deemed void or null, or be deemed legitimate, because of the mentioned incoherence between the SD and TSD.

3.3 Copyright

The purpose of this copyright section is to address what components of software that can be subject to copyright protection, what entailing rights the copyright holder has over the copyrighted matter, as well as how the copyright holder's rights relate to reverse engineering and decompilation.

3.3.1 Introduction

The overall purpose of the SD was to harmonize the protection of software programs, since it had been acknowledged that the inconsistent level of protection of software programs had a negative effect on the market, see recital 5 of the SD. Consequently, the directive was enacted in 1991 to harmonize the protection of software programs by giving software copyright protection as literary works within the meaning of the Bern Convention[90] see article 1(1) of the SD.[91] However, the directive did not define software programs further than stating that the definition included preparatory works.[92] The SD has since then been updated through Directive 2009/24/EC of the European Parliament and of the Council of 23 April 2009 on the legal protection of computer programs (Codified version). I will thus base my referrals on the articles and recitals of the latest version of the SD, unless stated otherwise.

3.3.2 Scope of Protection

The scope of protection of software is based on the idea-expression dichotomy codified in article 1(2) of the SD, which clearly states that expressions of software are protected, wherein ideas and underlying principles are not, including those underlying software interfaces.[93]

---

[90] The Bern Convention is an international convention regarding copyright which is administrated by the World Intellectual Property Organization (WIPO) see Levin. M. (2017). *Lärobok i immaterialrätt.* Stockholm: Wolters Kluwer Sverige AB, pp. 43 and 46.

[91] As a side note, it serves to mention the Directive 2001/29/EC of the European Parliament and of the Council of 22 May 2001 on the harmonisation of certain aspects of copyright and related rights in the information society, (InfoSoc), which was enacted 2001 with the purpose of harmonizing copyright and neighbouring rights in the EU, see recital 1 of the InfoSoc. Mentioned directive is however subsidiary to the SD and shall thus not affect the SD, see article 1(2)(a) of the InfoSoc.

[92] Dreier, T. (1991). *The Council Directive of 14 May 1991 on the Legal Protection of Computer Programs.* European Intellectual Property Review. 13(9) 319-330, p. 2, referring to Official Journal of the European Communities L 122 Volume 34. 17 May 1991, p. 43.

[93] The principle is referred to by the CJEU in Case C-406/10, *SAS Institute Inc. v. World Programming Ltd.,* paragraph. 52.

However, the described scope of protection was different when the directive was first proposed by the European Commission (Commission). The directive emerged from a heavy debate of lobbyism surrounding the question of whether *software interface* should be within the scope of copyrightable matter or not, which also entailed the question if reverse engineering and decompilation should be permissible.[94] The initial draft of the SD distinguished *software interface* from *software interface specifications,* wherein it was stated that copyright protection was not extended to ideas and principles underlying *software interface specifications.* Thus, the draft did not address the protection of *software interfaces*.[95] The proposal was heavily opposed, and the following debate led to a change of the draft.[96] The actors who opposed the draft, favored a narrow protection of software interfaces and advocated for a right to reverse engineer and decompile. Their reasoning was that their approach would facilitate interoperability of software programs and thereby favor the competition. Their opponents on the other hand reasoned that strong protection of software interfaces and protection against reverse engineering are needed in order to prevent software piracy and to favor innovation within the software industry.[97] Article 1(2) of the SD was influenced and altered by the arguments to explicitly state that ideas and principles, including those underlying *software interfaces*, are not subject to copyright protection.[98]

The scope of protection has since then been subject to interpretation. The CJEU addressed the protection of programming language and data file format in relation to article 1(2) of the SD in the *SAS Institute Inc. v. World Programming Ltd; C 406:10,* and concluded that such matters fall outside the scope of the article.[99] Whether that implies that software interface is possible to protect by copyright or not is currently unclear, and will not be elaborated further in this thesis.[100]

Copyright protection is furthermore dependent on the software program being original "in the sense that it is the author's own intellectual creation." according to article 1(3) of the SD.[101] What can be concluded is thus that the expression of software through source code and object code can be subject to copyright

---

[94] Palmer, A. K., Vinje, T. C. (1992). *The EC Directive on the legal protection of computer software: New law governing software development.* Duke Journal of comparative & international law, volume 2:65, 65-87, p. 68.
[95] Ibid, p. 71.
[96] Ibid.
[97] Ibid, pp. 69-71.
[98] Ibid, p. 78.
[99] Case C-406/10, *SAS Institute Inc. v. World Programming Ltd.,* paragraph. 39. On a side note it serves to be mentioned that the CJEU also stated that even though programming language and file format fall outside the scope of article 1(2) of the SD, mentioned matter may instead be copyrightable under the InfoSoc, see Case C-406/10, *SAS Institute Inc. v. World Programming Ltd.,* paragraph. 45.
[100] See further Wolk, S. (2016). *Datorprogramalster i upphovsrätten*. Uppsala: Iustus Förlag AB, p. 43, and Band, J. (2018). *The Global API Copyright Conflict.* Harvard Journal of Law & Technology, volume 31 Special Issue Spring 2018, 616-637, pp. 619-620.
[101] The originality requirement has been subject to several interpretations by the CJEU, see the noticeable and guiding Case C-5/08, *Infopaq International A/S v. Danske Dagblades Forening*, in which the requirement of originality is set quite low. Mentioned case addresses the InfoSoc which however relies on the same principle of originality as the SD, see Case C-5/08, *Infopaq International A/S v. Danske Dagblades Forening,* paragraph. 35-37. A lot of software programs are thus suggested to likely meet the requirement of originality, see further Wolk, S. (2016). *Datorprogramalster i upphovsrätten*. Uppsala: Iustus Förlag AB, p. 69 f.

protection provided that the requirement of originality is met (disregarding any potential copyright of software interfaces). Ideas and principles fall outside the scope of copyright, meaning that a machine learning algorithm in its abstract format cannot be protected by copyright.[102]

The following section will address what rights the copyright holder has over software components that are subject to copyright protection.

3.3.3 Copyright Holder's Exclusive Rights

The copyright protection gives the copyright holder the exclusive right to exercise or authorize certain acts relating to the copyrighted work in article 4 of the SD. The following acts are listed in article 4:

a) "the permanent or temporary reproduction of a computer program by any means and in any form, in part or in whole; in so far as loading, displaying, running, transmission or storage of the computer program necessitate such reproduction, such acts shall be subject to authorization by the right holder;

b) the translation, adaptation, arrangement and any other alteration of a computer program and the reproduction of the results thereof, without prejudice to the rights of the person who alters the program;

c) any form of distribution to the public, including the rental, of the original computer program or of copies thereof."

Therefore, the basis of the exclusive rights is that a user needs authorization from the copyright holder for running the software, see article 4(a) of the SD, or for translating (decompiling) the object code into source code, see article 4(b) of the SD. The exclusive rights can be said to be a monopoly over the copyrightable matter, in which the copyright holder determines the disposition of the copyrighted works.[103]

3.3.4 Reverse Engineering

However, the exclusive rights are restricted through the rights of reverse engineering and decompiling in article 5(3) and 6 of the SD. This and the next coming section will therefore address under what premises a user may perform the rights of reverse engineering and decompilation, and how the practices may be restricted by the copyright holder. Both rights of reverse engineering and decompilation will also be teleologically assessed in order to determine their underlying purpose.

---

[102] See Dreier, T. (1991). *The Council Directive of 14 May 1991 on the Legal Protection of Computer Programs.* European Intellectual Property Review. 13(9) 319-330, p. 2, referring to Official Journal of the European Communities L 122 Volume 34. 17 May 1991, p. 43.

[103] See Boyle, J. (2003). *The second enclosure movement and the construction of the public domain*. Law and Contemporary Problems, 33-74, pp. 54-55.

As for reverse engineering, article 5(3) of the SD covers the right to reverse engineer and is formulated as follows:

> "The person having a right to use a copy of a computer program shall be entitled, without the authorization of the rightholder, to observe, study or test the functioning of the program in order to determine the ideas and principles which underlie any element of the program if he does so while performing any of the acts of loading, displaying, running, transmitting or storing the program which he is entitled to do."

Article 8(2) of the SD relates to article 5(3) of the SD, and states that: "Any contractual provisions contrary to Article 6 or to the exceptions provided for in Article 5(2) and (3) shall be null and void.", and is thus imposing that article 5(3) is compulsory.

Based on the formulation of article 5(3) of the SD, one can define the right to reverse engineer as follows:

> A right to "observe, study or test the functioning of the program in order to determine the ideas and principles which underlie any element of the program […]" without a need of authorization from the copyright holder, see article 5(3) of the SD.[104]

Furthermore, article 5(3) stipulates two requirements for exercising the right to reverse engineer, namely:

i) The user shall have a right to use a copy of the software, and;

ii) Reverse engineering is exercised whilst doing "any of the acts of loading, displaying, running, transmitting or storing the program which he is entitled to do.*"*[105]

The CJEU addressed the meaning of requirement ii), "any of the acts of loading, displaying, running, transmitting or storing the program which he is entitled to do.*"* in the *SAS Institute Inc. v. World Programming Ltd case*, wherein CJEU stated that the acts may be performed "within the framework of the acts permitted by the license." as a basis.[106] The CJEU continued its reasoning by referring to the Advocate General's opinion on the matter who stated: *"*It is my view that the expression 'any of the acts of loading, displaying, running, transmitting or storing the computer program [which the person having

[104] Chosen definition is influenced by how Dreier chose to define article 5(3) of the SD, see Dreier, T. (1991). *The Council Directive of 14 May 1991 on the Legal Protection of Computer Programs.* European Intellectual Property Review. 13(9) 319-330, p. 5.
[105] Article 5(3) of the SD.
[106] Case C-406/10, *SAS Institute Inc. v. World Programming Ltd.*, paragraphs. 54-55, and Opinion of Advocate General BOT Delivered on 29 November 2011; Case C-406/10 SAS Institute Inc. v World Programming Ltd., paragraph. 94.

the right] is entitled to do' relates to the acts authorized under Articles 4(a) and (b) and 5(1) of Directive 91/250."[107]

From the foregoing section, we know that Article 4(a) and (b) of the SD are acts within the exclusive rights of the copyright holder. As for article 5(1), the article stipulates that a lawful user shall not need authorization from the copyright holder for performing the acts of article 4(a) and (b) of the SD when they are necessary for the use of the software in accordance to the software's purpose and for performing error correction, *unless* there is a contract stating otherwise.[108] Thus, article 5(1) of the SD is enabling contractual restrictions of its referred acts under article 4(a) and (b) of the SD.[109] However, the CJEU as well as the Advocate General, state that the possibility to contractually restrict the referred acts under article 5(1) is limited by recital 17 of Directive 91/250, now recital 13 of the SD, which stipulates that "the acts of loading and running necessary for the use of a copy of a program […] may not be prohibited by contract."[110]

Consequently, the copyright holder holds the right to authorize "any of the acts of loading, displaying, running, transmitting or storing the program which he is entitled to do." through e.g. a license agreement.[111] However, the "acts of loading and running necessary for the use of a copy of a program […]" cannot be restricted and shall be allowed independently of what is stated in a license agreement, (a limitation of the copyright holder's exclusive disposition right).[112]

As a last remark, I would like to refer to the Advocate General's opinion which was largely reflected in how the CJEU chose to interpret the case.[113] The Advocate General stated that the right to reverse engineer should be allowed to the extent the exclusive rights of the copyright holder are not neglected, which further underpins that it is the right holder, i.e. licensor, which determines which acts that may be performed by the user for performing reverse engineering – *apart* from the "acts of loading and running necessary for the use of a copy of a program […]"[114]

---

[107] Case C-406/10, *SAS Institute Inc. v. World Programming Ltd.*, paragraph. 57, and Opinion of Advocate General BOT Delivered on 29 November 2011; Case C-406/10 SAS Institute Inc. v World Programming Ltd., paragraph. 95.
[108] Article 5(1) of the SD.
[109] See further Opinion of Advocate General BOT Delivered on 29 November 2011; Case C-406/10 SAS Institute Inc. v World Programming Ltd., paragraph. 96.
[110] Case C-406/10, *SAS Institute Inc. v. World Programming Ltd.*, *p*aragraph 58, and Opinion of Advocate General BOT Delivered on 29 November 2011; Case C-406/10 SAS Institute Inc. v World Programming Ltd., paragraph. 96.
[111] See Article 5(3) of the SD, and Case C-406/10, *SAS Institute Inc. v. World Programming Ltd.,* paragraph 54-55.
[112] Case C-406/10, *SAS Institute Inc. v. World Programming Ltd.*, paragraph. 58, and recital 13 of the SD. It serves to be mentioned that recital 13 does not solely motivate the interpretation, since the CJEU refers to the recital in their preliminary ruling, thus giving the interpretation legal authority.
[113] See CJEU's referrals in Case C-406/10, *SAS Institute Inc. v. World Programming Ltd*., e.g. paragraph. 57.
[114]Opinion of Advocate General BOT Delivered on 29 November 2011; Case C-406/10 SAS Institute Inc. v World Programming Ltd., paragraph. 99.

The following illustration aims to visualize the contractual scope of limiting the right to reverse engineer in relation to the copyright holder's exclusive rights, i.e. the copyright holder's possibility to restrict reverse engineering in article 5(3) of the SD.

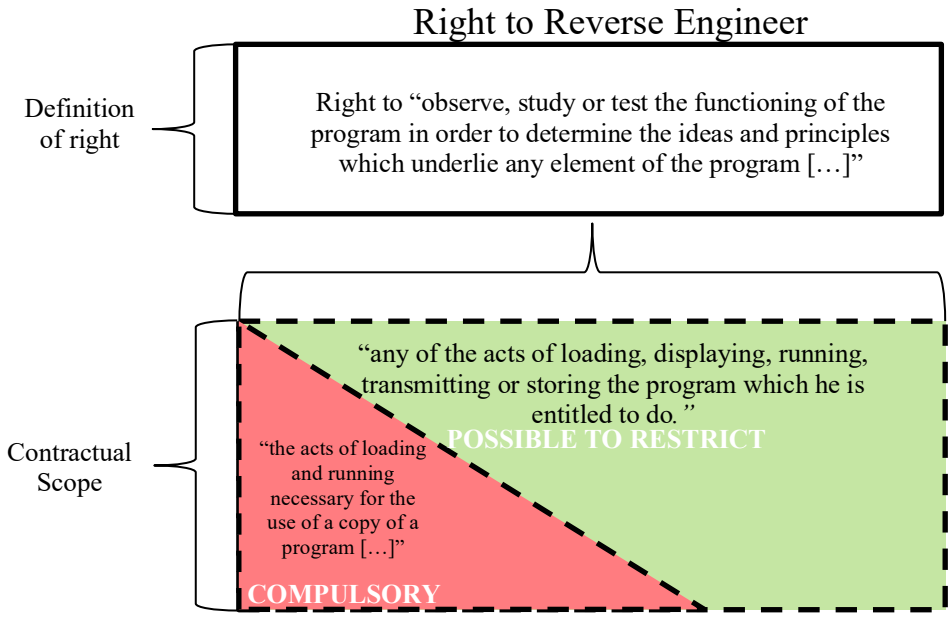

Figure 2: Illustration of the contractual scope for limiting reverse engineering in relation to the right holder's exclusive rights, illustrated by me.[115]

3.3.4.1 Purpose of article 5(3) of the SD

Articles 5(3) and 6 of the SD was not included in the first draft of the directive, and the articles was first included after strong pressure had been put on the Commission by companies who opposed the draft on the basis of wanting to enable reverse engineering in favor of interoperability.[116]

The purpose of article 5(3) has been addressed further by the CJEU in the same case of *SAS Institute Inc. v. World Programming Ltd* by stating the two following paragraphs:

i) "In this respect, Article 5(3) of Directive 91/250 seeks to ensure that the ideas and principles which underlie any element of a computer program are not protected by the owner of the copyright by means of a licensing agreement."[117]

ii) "That provision [Article 5(3) of the SD] is therefore consistent with the basic principle laid down in Article 1(2) of Directive 91/250, pursuant to which protection in accordance with that

[115] Illustration is based on article 5(3) of the SD, Case C-406/10, *SAS Institute Inc. v. World Programming Ltd.,* paragraphs. 54-55 and 58, and recital 13 of the SD.
[116] Palmer, A. K., Vinje, T. C. (1992). *The EC Directive on the legal protection of computer software: New law governing software development.* Duke Journal of comparative & international law, volume 2:65, 65-87, pp.71-72 and 74.
[117] Case C-406/10, *SAS Institute Inc. v. World Programming Ltd.*, paragraph. 51.

directive applies to the expression in any form of a computer program and ideas and principles which underlie any element of a computer program are not protected by copyright under that directive."[118]

Thus, the purpose of article 5(3) of the SD is clearly stated by the CJEU to prevent the copyright holder from gaining control over ideas and principles that are not subject to copyright according to article 1(2) of the SD through the means of a license agreement, see i) above. The Advocate General elaborates further on paragraph ii) by stating that 5(3) is an extension of 1(2), which indicates that the interests of interoperability underlying 1(2) are relevant for article 5(3).[119]

The underlying interoperability interests of the SD were also confirmed by the Commission in a Commission Staff Working Document which analyzed the need for having an interoperability directive.[120] The Commission stated that both article 5(3) relating to reverse engineering and article 6 relating to decompilation had the purpose of promoting interoperability.[121]

One may thus conclude that article 5(3) of the SD aims to prevent an extension of the copyright protection through a license agreement in order to comply with the idea-expression dichotomy codified in article 1(2) of the SD.

3.3.5 Decompilation

Article 6 of the SD covers the right to decompile software, i.e. translate object code into source code, and can be said to be a narrower right than the right to reverse engineer in 5(3) of the SD since decompilation is only allowed for achieving interoperability between one software and another, which is not the case for reverse engineering in article 5(3) of the SD.[122]

Article 6(1) of the SD is formulated as follows:

"The authorisation of the rightholder shall not be required where reproduction of the code and translation of its form within the meaning of points (a) and (b) of Article 4(1) are indispensable to obtain the information necessary to achieve the interoperability of an independently created computer program with other programs, provided that the following conditions are met:

---

[118] Ibid, paragraph. 52.
[119] Opinion of Advocate General BOT Delivered on 29 November 2011; Case C-406/10 SAS Institute Inc. v World Programming Ltd., paragraph. 92.
[120] European Commission Staff Working Document Analysis of measures that could lead significant market players in the ICT sector to license interoperability information. SWD(2013) 209 final, pp. 12-13.
[121] Ibid, pp. 11 and 13.
[122] See Shemtov, N. (2017). *Beyond the Code Protection of Non-Textual Features of Software.* Oxford: Oxford University Press, p. 85.

a) those acts are performed by the licensee or by another person having a right to use a copy of a program, or on their behalf by a person authorised to do so;
b) the information necessary to achieve interoperability has not previously been readily available to the persons referred to in point (a); and
c) those acts are confined to the parts of the original program which are necessary in order to achieve interoperability."

The sentence "information necessary to achieve the interoperability of an independently created computer program with other programs[…]" in article 6(1) of the SD, is regarded as software interfaces by Dreier, and I agree with such an interpretation considering the definition of software interfaces in recital 10 of the SD.[123]

Based on the formulation of article 6(1) of the SD, one should thus be possible to define the right to decompile as follows:

> The user does not need an authorization from the copyright holder to perform the acts of "reproduction of the code and translation of its form within the meaning of points (a) and (b) of Article 4(1) […]", provided that mentioned acts are indispensable to obtain software interfaces.[124]

Furthermore, in legal literature it is suggested that decompilation is dependent on the acts being indispensable which imposes that a user may not decompile without authorization from the right holder if the same information can be acquired through reverse engineering according to article 5(3) of the SD or if the information previously has been made available through e.g. a document specifying the information needed for achieving interoperability.[125] One could thus argue that the prerequisite of *indispensability* in article 6(1) of the SD, enables an interpretation in which one should be possible to contractually restrict *non-indispensable* acts of decompilation.[126]

The right to decompile is also dependent on the fulfilment on the requirements according to article 6(1)(a)-(c) of the SD, and the information that has been obtained through decompilation is as well

[123] Article 6(1) of the SD, and Dreier, T. (1991). *The Council Directive of 14 May 1991 on the Legal Protection of Computer Programs.* European Intellectual Property Review. 13(9) 319-330, p. 7.
[124] See Shemtov, N. (2017). *Beyond the Code Protection of Non-Textual Features of Software*. Oxford: Oxford University Press, p. 85, and Dreier, T. (1991). *The Council Directive of 14 May 1991 on the Legal Protection of Computer Programs.* European Intellectual Property Review. 13(9) 319-330, p. 7, both in which the scope of the right to decompile is addressed.
[125] Dreier, T. (1991). *The Council Directive of 14 May 1991 on the Legal Protection of Computer Programs.* European Intellectual Property Review. 13(9) 319-330, p. 7, and Shemtov, N. (2017). *Beyond the Code Protection of Non-Textual Features of Software.* Oxford: Oxford University Press, p. 85.
[126] See Ibid.

restricted through the second paragraph of the article, 6(2) of the SD. Article 6(2) of the SD states that the information shall not be used for other purposes than:

(i) achieving interoperability with an independently created software,
(ii) be transferred to persons unless it is required for achieving interoperability,
(iii) be used for developing, producing or marketing a software which is substantially similar to the protected software or;
(iv) Be used for actions that constitute copyright infringement.

It does also serve to be mentioned that even though the usage of information obtained through decompilation is restricted in article 6(2) of the SD, the article does not impose a positive right to use the information.[127]

The final paragraph of article 6(3) of the SD aims to protect the interests of the copyright holder, stating that the article should not be interpreted in a way that unreasonably neglects the interests of the copyright holder or goes beyond a normal use of a software.

Hence, the right to decompile is quite explicitly describing the premises for acquiring software interface information, and for which purposes the information may be used. If the premises are not respected and the user acts without authorization, there will be a question of copyright infringement, see article 6(1) of the SD in comparison with article 4(1) of the SD.

Similar to article 5(3), decompilation of article 6 is compulsory due to article 8(2) of the SD, which imply that the indispensable acts of "reproduction of the code and translation of its form within the meaning of points (a) and (b) of Article 4(1) […]" for obtaining software interface information in relation to the requirements in 6(1)(a)-(c), may not be contractually restricted by the copyright holder considering article 8(2) of the SD.

3.3.5.1 Purpose of article 6 of the SD

The purpose of article 6 of the SD is quite explicit about promoting interoperability since the right only admits acquisition and use of information when it is indispensable for creating an interoperable software.[128]

The right to decompile and the protection of *software interfaces* was intertwined in the debate surrounding the legislative procedure of the SD since the right to decompile relates to obtaining *software*

[127] Shemtov, N. (2017). *Beyond the Code Protection of Non-Textual Features of Software*. Oxford: Oxford University Press, p. 98.
[128] Ibid, p. 85.

interface information. The reason for introducing a right to decompile and exclude ideas and principles underlying software interfaces was to prevent monopolization of functional aspects of software which otherwise could lead to lock-in effects and harm the competition.[129] The change in scope of 1(2) of the SD and the inclusion of the right to decompile software for obtaining interfaces was thus a response to said concerns.[130]

My understanding is thus that the copyright framework does not acknowledge an intrinsic value in decompilation as such, but aims to secure that copyright protection does not impose monopolization of ideas and principles including functional aspects of software.[131]

3.4 Trade Secret

Similar to the copyright section, this trade secret section has the purpose of addressing what components of software that can be subject to trade secret protection, what entailing protection the trade secret holder has over the trade secret protected matter, and how reverse engineering (including decompilation) relates to the trade secret protection.

3.4.1 Introduction

Trade secret protection is regulated in the EU through the TSD. The TSD's overall purpose can be said to be harmonization and to introduce a minimum protection for trade secrets in the EU, in order to promote cross-border collaboration and to strengthen trade secrets from a competition point-of-view.[132] The directive was proposed after several reports concluded that the inconsistent framework for trade secrets imposed uncertainty, high transaction costs and competition issues for companies relying on trade secrets.[133]

3.4.2 Scope of Protection

The subject matter for trade secret protection is information, see article 2(1) of the TSD. The directive does however not specify the meaning of information, but it has been addressed in the preparatory work of the Swedish Trade Secret Act (2018:558) (TSA) which incorporates the TSD. The definition of information is said to have a broad meaning and shall be equivalent to the use of the concept in normal

[129] Palmer, A. K., Vinje, T. C. (1992). *The EC Directive on the legal protection of computer software: New law governing software development.* Duke Journal of comparative & international law, volume 2:65, 65-87, pp. 69-70.
[130] Ibid, pp. 69-72 and 74.
[131] See Case C-406/10, *SAS Institute Inc. v. World Programming Ltd.*, paragraph. 40, in which the CJEU addresses the importance of not monopolizing functional aspects of software.
[132] TSD recital 3, and SOU 2017:45. En ny lag om företagshemligheter, pp. 79-80.
[133] SOU 2017:45. En ny lag om företagshemligheter, p. 79.

speech. The medium of the information is irrelevant which makes the concept of information include information that is digitally stored. Source code and software have been specifically mentioned in Swedish preparatory works as matters that are included in the definition.[134] Hence, I will make an assumption that source code and object code are to be considered potential trade secret protected matter in this thesis.

The TSD sets up three requirements for information to be deemed as trade secrets, which all need to be fulfilled in order for the trade secret protection to apply, see article 2(1) of the TSD which is based on article 39 of the TRIPS-agreement.[135] The requirement for protection in article 2(1) of the TSD are as follows:

Information needs to be:

(a) Secret, which means that the information is not publicly known in its entirety or in the form its parts have been arranged and put together or easy to obtain for persons that normally have access to mentioned type of information.
(b) Of commercial value because it is kept secret
(c) Subject to reasonable actions for keeping the information secret

It may be clarified that only putting a software application on the market without the source code does not impose that the requirement of secrecy for obtaining trade secret protection has been met according to article 2(1) of the TSD.[136] The situation of this thesis is however focusing on a license relationship, in which the requirements for protection in article 2(1) of the TSD can be met through contractual provisions such as confidentiality clauses.[137]

Conclusively, all of the components of software, including abstract information may thus be subject to trade secret protection provided that all the requirements of article 2(1) of the TSD are met.

3.4.3 Right and purpose of Reverse Engineering

In recital 16 of the TSD, it is clearly stated that trade secrets shall not be considered as an exclusive right to the protected information with the purpose of promoting innovation and competition. Trade secrets

[134] Prop 2017/18:200. En ny lag om företagshemligheter, p. 138.
[135] Annex 1C of the Marrakesh Agreement Establishing the World Trade Organization, signed in Marrakesh, Morocco, on 15 April 1994, Article 39.
[136] See Dreier, T. (1991). *The Council Directive of 14 May 1991 on the Legal Protection of Computer Programs.* European Intellectual Property Review. 13(9) 319-330, footnote. 61.
[137] See Fahlbeck, R. (2019). *Lagen om skydd för företagshemlighet En kommentar och rättsöversikter.* Stockholm: Norstedt Juridik AB, p.388.

do therefore not have the same standing as intellectual property rights.[138] Mentioned standing is also clarified by article 2 of the TSD which states that one can hold but mentions nothing about owning a trade secrets which is the case for other intellectual property rights which one owns.[139]

It does however serve to be mentioned that reverse engineering in the TSD is not solely motivated by recital 16, but also in relation to article 3(1)(a) and (b) of the TSD, article which will be addressed in the next section.

The right to reverse engineer can be said to be a consequence of the characteristic of the trade secret protection itself. The Commission clarifies the reasoning by stating that exclusivity of information is given by other intellectual property rights, and that reverse engineering of trade secrets shall be permissible in order to allow for distinguishing the mere acquisition of knowledge from misappropriation of information.[140] Thus, it is present that the trade secret directive is positive to reverse engineering as such.

The next coming section will address the protection held by a trade secret holder, i.e. what constitute lawful and prohibited dispositions of trade secrets, with the purpose shedding light on how reverse engineering relates to trade secret protection.

3.4.4 Trade Secret Protection

3.4.4.1 Lawful Acquisitions

Article 3(1) of the TSD describes lawful acquisitions of trade secrets, wherein four types of acquisitions are deemed lawful, namely:

a) "Independent discovery or creation;
b) observation, study, disassembly or testing of a product or object that has been made available to the public or that is lawfully in the possession of the acquirer of the information who is free from any legally valid duty to limit the acquisition of the trade secret;
c) exercise of the right of workers or workers' representatives to information and consultation in accordance with Union law and national laws and practices;

[138] SOU 2017:45. En ny lag om företagshemligheter, pp. 80-81.
[139] Ibid.
[140] European Commission Staff Working Document Impact Assessment *Accompanying the document* proposal for a Directive of the European Parliament and of the Council on the protection of undisclosed know-how and business information (trade secrets) against their unlawful acquisition, use and disclosure. SWD(2013) 471 final, pp. 45 and 246.

d) any other practice which, under the circumstances, is in conformity with honest commercial practices."

Article 3(1)(a) of the TSD codifies that information protected as trade secrets are not exclusive and may be acquired by others through independent discovery or creation, which reconnects to recital 16 of the TSD and gives an explanation why reverse engineering is permissible as a basis.

Article 3(1)(b) of the TSD does also relate to recital 16, wherein both the article and recital legitimize reverse engineering as a lawful action for acquiring trade secrets. Article 3(1)(b) is requiring that the product or object containing the trade secret has been made available to the public or that the acquirer has the product or object in her possession lawfully, through e.g. a license, and is also stating that the acquirer shall be "free from any legally valid duty to limit the acquisition […]", which must refer to the possibility to contractually limit reverse engineering given by recital 16 of the TSD.[141] Article 3(1)(c) of the TSD is not relevant for this thesis and will not be addressed further.

Article 3(1)(d) of the TSD is a general clause enabling a freer interpretation of what constitute a lawful acquisition based on the specific circumstances of the case. The paragraph refers to honest commercial practices which is specified in the TRIPS-agreement wherein the following definition is given: "'a manner contrary to honest commercial practices' shall mean at least practices such as breach of contract, breach of confidence and inducement to breach, and includes the acquisition of undisclosed information by third parties who knew, or were grossly negligent in failing to know, that such practices were involved in the acquisition."[142]Article 3(1)(d) of the TSD is thus also enabling a restriction of reverse engineering through contracts.

3.4.4.2 Prohibited Acquisitions, Use and Disclosure

The directive divides unlawful actions into acquisitions, use and disclosure, wherein article 4(2) of the TSD cover four types of unlawful acquisitions of trade secrets, namely:

i) unauthorized access to,
ii) appropriation of, or;
iii) copying of, any documents, objects etc. that contain the trade secret or from which the trade secret can be deduced, or;
iv) by any other conduct which, under the circumstances, is considered contrary to honest commercial practices.[143]

---

[141] SOU 2017:45. En ny lag om företagshemligheter, p. 119.
[142] Annex 1C of the Marrakesh Agreement Establishing the World Trade Organization, signed in Marrakesh, Morocco, on 15 April 1994, Article 39 and footnote 10, and Clark, B., Martinis, L. D., & Niebel, R. (2018). *The EU Trade Secrets Directive: all change for trade secret protection in Europe?* Journal of Intellectual Property Law & Practice, vol. No. 6, 445-457, pp. 449-450.
[143] SOU 2017:45. En ny lag om företagshemligheter, p. 130.

Honest commercial practices is once again referring to the definition in the TRIPS-agreement, wherein an acquisition of trade secrets in breach of contracts are deemed unlawful.[144] It is thus apparent that a trade secret holder, i.e. a licensor, may stipulate in a contractual provision that acquisition of a trade secret is prohibited for a contracting party, thus making an acquisition of such unlawful according to the (iv) requirement of 4(2) of the TSD.[145]

Article 4(3) stipulates what an *unlawful use and disclosure* of trade secret is:

> ”The use or disclosure of a trade secret shall be considered unlawful whenever carried out, without the consent of the trade secret holder, by a person who is found to meet any of the following conditions:
>
> a) having acquired the trade secret unlawfully;
> b) being in breach of a confidentiality agreement or any other duty not to disclose the trade secret;
> c) being in breach of a contractual or any other duty to limit the use of the trade secret.”

Article 4(3)(b) and (c) are thus enabling the trade secret holder to both restrict disclosure and use of trade secret through contractual provisions for the trade secret acquirer, i.e. a licensee, through contractual provisions.

Article 4(4) and 4(5) of the SD will not be addressed since the focus of this thesis lie on acquisition and use of trade secrets within a license relationship.

3.4.4.3 Conclusion

Consequently, if a licensee breaches a contractual provision to *acquire* a trade secret, there has been an unlawful *acquisition* of trade secrets according to article 3(1) (b) and (d) and 4(2) of the TSD. Breaching a contractual provision by *using* a trade secret outside of the permitted scope will be an unlawful use of a trade secret according to 4(3)(b) and (c) of the TSD.

3.5 Incoherence between the SD and TSD in legal literature

This section will address how the problem statement, the incoherence between the SD and TSD, has been interpreted in legal literature. I will thus describe the authors’ views in order to nuance the problem statement of this thesis.

---

[144] Clark, B., Martinis, L. D., & Niebel, R. (2018). *The EU Trade Secrets Directive: all change for trade secret protection in Europe?* Journal of Intellectual Property Law & Practice, vol. No. 6, 445-457, pp. 449-450.
[145] See Ibid.

In the above mentioned book *Beyond the Code: Protection of Non-Textual Features of Software* by Shemtov, the incoherence between the TSD and SD is addressed.[146] Shemtov states that if a licensee chooses to reverse engineer or decompile a licensed software to *acquire* a trade secret although an action of such is in breach with an anti-reverse engineering clause, the action cannot be deemed to be an *unlawful trade secret acquisition* according to article 3(1)(b) and 4(2) of the TSD since the contractual provision would be ineffective due to being in breach with article 5(3), 6 and 8(2) of the SD.[147]

Shemtov continues his reasoning by also addressing the possibility to restrict *usage* of information obtained through reverse engineering and decompilation in his publication. Shemtov says that article 8(1) of the SD which states that the SD shall not affect other regulations including those concerning trade secrets, can enable an interpretation in which a contractual clause can restrict *usage* of information obtained from reverse engineering by relying on the TSD instead of the SD.[148] In other words, if a licensee would obtain information through reverse engineering, and then *use* it outside the scope for what is permitted by the confidentiality clause, there would be an unlawful *use* of a trade secret according to article 4(3)(b) of the TSD. However, Shemtov later claims that such a contractual clause would likely be denied by the CJEU because of their reasoning presented in the *SAS Institute Inc. v. World Programming Ltd* case. Shemtov thinks that the CJEU will consider the pro-competition policy that underlies article 5(3), 6 and 8 of the SD, and consider the overall negative impact that such a clause will have on competition, and thus conclude that mentioned clause will be ineffective.[149]

Schröder has also addressed the incoherence between the directives and states that the SD seems to be an existing limitation of the TSD. Schröder underpins his reasoning by referring to recital 16 of the TSD which states that the right to conclude contracts that restrict reverse engineering can be limited by law. The wording limitation by "law" in mentioned recital should thus be considered as a reference to the compulsory reverse engineering rights of the SD, imposing that contractual restrictions in contrary to mentioned rights will be deemed null according to article 8(2) of the SD.[150]

Dreier addressed the SD in a comment on the directive back in 1991, where Dreier states that the article 8(1) of the SD enables an interpretation that the compulsory character of decompilation does not apply

[146] Shemtov, N. (2017). *Beyond the Code Protection of Non-Textual Features of Software.* Oxford: Oxford University Press, Chapter 1 and 8.
[147] Ibid, p. 230.
[148] Ibid.
[149] Ibid.
[150] See Schröder, V. (2018). *Transformation from a Patchwork Quilt to a Unified Fabric: Discussion on a Few Particular Aspects of the New Finnish Trade Secrets Act and the EU Trade Secrets Directive.* Nordiskt Immateriellt Rättsskydd, NIR 4 / 2018, 500-515, p. 507.

to trade secret protection.[151] Dreier does however nuance his interpretation by stating that having contractual provisions which rely on trade secret protection that prevents *usage* of *software interface information* would contradict the underlying purpose of the directive – to secure access *to software interfaces* for interoperability purposes. His conclusion is therefore that *software interface information* that has been obtained in compliance with the decompilation right in article 6 of the SD cannot be contractually restricted in regards of *usage* by relying trade secret protection.[152] It does serve to be mentioned that when Dreier published his comment, there were only national regulations on trade secret protection and the TSD had not yet been adopted. His reasoning is thus not addressing the dissonance between the SD and TSD per se, but is relevant for shedding light on the purpose of the SD and how the directive relates to trade secrets.

[151] Dreier, T. (1991). *The Council Directive of 14 May 1991 on the Legal Protection of Computer Programs.* European Intellectual Property Review. 13(9) 319-330, p. 8.
[152] Ibid.

4 Analysis

4.1 Introduction

This chapter has the purpose of answering the RQ of this thesis by addressing each sub question, RQ(a)-(c). Lastly this chapter will be concluded with some closing words.

4.2 RQ(a) How do the TSD and SD interrelate with software?

4.2.1 Combined Scope of Protection

From the copyright section of this thesis, it is clarified that copyright protection can apply to the expressions of the software, provided that the requirement of originality is met in article 1(3) of the SD. Hence, copyright protection can apply to object code and source code, and not abstract information such as the machine learning algorithm in its abstract format, see article 1(2) of the SD. Whether copyright protection applies to software interfaces or not was not concluded due to a delimitation of this thesis.

In regards of trade secret protection, all parts of the software can be subject to protection provided that the requirements of protection are met in the TSD. Object code and source code, including software interface, as well as abstract information such as the machine learning in its abstract format can therefore be subject to trade secret protection, see article 2(1) of the TSD.

Consequently, the combined scope of protection of copyright and trade secret for Med-Company Z's software are intersecting in source and object code, but not in the machine learning algorithm in its abstract format since only trade secret protection applies to such information. Whether copyright and trade secrets intersect regarding software interfaces is left unanswered in this thesis because of above mentioned delimitation. The next coming sections will therefore continue with disregarding any potential copyright protection of software interfaces.

4.2.2 Combined Rights and Control

From a mere copyright perspective, Med-Company Z can contractually authorize or restrict dispositions of the source and object code, but has no right to authorize or restrict disposition of the algorithm in its abstract format. The possibility to authorize or restrict dispositions over copyrightable matter is furthermore restricted through the rights of reverse engineering in 5(3) and decompilation in article 6 of the SD. The consequence for a user not complying with mentioned authorization or restriction, is a copyright infringement by the user of the software as well as a breach of contract, see article 4 of the SD.

From a mere trade secret perspective, Med-Company Z can contractually restrict the acquisition of the components of the software including the machine learning in its abstract format which impose that an acquisition in breach with such a contractual provision will be considered an unlawful acquisition of trade secrets according to article 3(1)(b) and (d) and 4(2) of the TSD. Med-Company Z can as well restrict the use of the software components including the machine learning algorithm in its abstract format, wherein a breach of such a provision constitute an unlawful use of trade secret according to article 4(3(b) and (c) of the TSD.[153]

Consequently, Med-Company Z can rely on both copyright and trade secret protection for controlling the source and object code, but the machine learning algorithm in its abstract format can only be protected as a trade secret. However, since abstract information is embedded in the software's object and source code, a user will still need to analyze the copyrighted expressions of the software in order to obtain the underlying abstract information.[154] A user must thus relate to the copyright regulation of the SD when performing reverse engineering or decompilation even though the objective is to obtain ideas and principles not protected by copyright. Reverse engineering and decompilation of trade secrets in software will thus actualize both the SD and TSD.

4.2.3 Conclusion

My answer to RQ(a) is that the combined scope of protection of copyright and trade secret are intersecting in source and object code, but not in the abstract information. Hence, the software owner can rely on both copyright and trade secret protection for controlling source and object code, but abstract information can only be protected as a trade secret. Nevertheless, reverse engineering and decompilation

[153] See Shemtov, N. (2017). *Beyond the Code Protection of Non-Textual Features of Software.* Oxford: Oxford University Press, p. 230.
[154] See Figure 1, wherein it is visualized that abstract information is behind the expressions of object code and source code.

of trade secrets in software will actualize both the SD and TSD since the practices relate to both copyright and trade secret protected matter.

4.3 RQ(b) Should one rely on the TSD or SD in regards of reverse engineering?

4.3.1 Introduction

The problem statement of this thesis sheds light on the fact that the TSD and SD has two different standings on whether contractual restrictions of reverse engineering and decompilation should be permissible or not, and that each directive states that it shall not affect the other. The sub question of RQ(b) will thus address which one of these two directives that should have priority over the other, in regards of the permissibility of contractual restrictions of reverse engineering and decompilation.

Relying on the TSD's regulations will have the effect of giving the software owner, e.g. Med-Company Z, an unlimited right to restrict reverse engineering and decompilation, whereas relying on the SD will imply a non-existent or limited right to restrict reverse engineering and decompilation because of the SD's article 5(3), 6 and 8(2).[155] The two next coming sections will elaborate on arguments for and against relying on the TSD as well as the SD.

4.3.2 Arguments for relying on the TSD

One can argue that one should rely on the TSD based on the following grounds combined:[156]

i) Exception I in the TSD which allows restriction of reverse engineering through contracts, and
ii) Article 8(1) of the SD which states that "The provisions of this Directive shall be without prejudice to any other legal provisions such as those concerning […] trade secrets".

The permissibility of contractual restrictions can be backed up by accentuating the intellectual property law's importance for giving control/exclusivity on the market.[157] The control/exclusivity can be said to promote innovation for innovators by enabling a lead time on the market which makes it possible to

[155] Compare Shemtov, N. (2017). *Beyond the Code Protection of Non-Textual Features of Software.* Oxford: Oxford University Press, pp. 230-232 where Shemtov elaborates on different interpretations of the incoherence.
[156] See Schröder, V. (2017, January). *Decompiling the rules on trade secrets, software and reverse engineering*. Downloaded 2019-02-03, from https://trustinip.com/decompiling-the-rules-on-trade-secrets (website no longer available), and Shemtov, N. (2017). *Beyond the Code Protection of Non-Textual Features of Software.* Oxford: Oxford University Press, p. 232.
[157] See Kungl. IngenjörsVetenskapsAkademien (IVA). (2011). *Immateriella Rättigheter och Ekonomiska Incitament En kortfattad översikt av kunskapsläget.* Stockholm: Kungl. Ingenjörsvetenskapsakademien (IVA), p. 6.

recoup R&D expenses.[158] One can of course argue that innovators already are protected by lead time on the market due to the costs and time connected to reverse engineering.[159] However, the relevance of such an argument can be questioned, since software is both quite cheap to produce and knowledge-intensive, implying that competitors can significantly reduce such a lead time through reverse engineering.[160]

Furthermore, exclusivity/control is closely connected with the interest of turnover, since control of know-how and software facilitates transaction models such as different licensing models in which information can be shared in exchange of financial capital.[161] Continuing on the same note, if the possibility to restrict reverse engineering of trade secrets in software does not exist, it will be more beneficial to transact trade secrets separated from software, than implemented in software, which is an odd consequence that may cause increased transaction costs which negatively affect the competition and incentives to innovate.[162]

4.3.3 Arguments for relying on the SD

A counterargument which instead supports an interpretation that the permissibility of contractual restrictions should be addressed in relation to the SD can also be found in recital 16 of the TSD, since mentioned recital stipulates that the possibility to contractually restrict reverse engineering can be restricted in itself by law (Exception II in the TSD). One can thus argue that article 5(3) and 6 in relation to 8(2) of the SD constitute such a restriction in "law", implying that contractual provisions relating to reverse engineering must comply with the SD.[163] Relying on the SD can also be backed up by recital 39 of the TSD, which states that the TSD shall not affect other intellectual property regulations.

Furthermore, one can also argue that relying on the TSD enables an unlimited restriction of reverse engineering which is questionable from a teleological standpoint, since the rights of reverse engineering and decompilation in the SD would be made completely irrelevant if one can restrict reverse engineering freely. And considering the CJEU's tendency to teleologically interpret directives – it is most likely difficult to circumvent the rights given by the SD by claiming that TSD enables a free restriction of

---

[158] See Samuelsson, P. Scotchmer, S. (2002). *The Law and Economics of Reverse Engineering.* The Yale Law Journal, volume 111:1575, 1575–1663, p. 1582.
[159] Ibid.
[160] See Davis, R., Kapor, M. D., Reichman, J. H., & Samuelsson. (1994). *A Manifesto concerning the Legal Protection of Computer Programs.* Columbia Law Review, Vol. 94:2308. 2308-2431, p. 2333.
[161] See Kungl. IngenjörsVetenskapsAkademien (IVA). (2011). *Immateriella Rättigheter och Ekonomiska Incitament En kortfattad översikt av kunskapsläget.* Stockholm: Kungl. Ingenjörsvetenskapsakademien (IVA), p. 6.
[162] See arguments for harmonizing trade secret protection through the TSD, described in section 3.3.1 in this thesis.
[163] Schröder, V. (2018). *Transformation from a Patchwork Quilt to a Unified Fabric: Discussion on a Few Particular Aspects of the New Finnish Trade Secrets Act and the EU Trade Secrets Directive.* Nordiskt Immateriellt Rättsskydd, NIR 4 / 2018, 500-515, p. 507.

reverse engineering.[164] The same interpretation can also be said to be compliant with the opinions expressed in legal literature.[165]

Moreover, the software industry is unlike other industries highly dependent on interoperability for having a fair competition since software applications rarely operate without interaction with other applications and platforms.[166] Having access to software interfaces is thus essential for making software interoperable and relevant on the market.[167] If Med-Company Z's software would not run on Windows or any other popular operating system because the software is lacking the necessary software interfaces, its relevance would certainly decrease. Hence, restricting reverse engineering might cause a negative impact on the competition since new actors risk being shut out from the market if they do not have access to necessary software interfaces.

One can also question if control of know-how is such a prominent driver for innovation that it is proclaimed to be, or if such an interpretation is primarily favored by larger software companies wanting to defend their market position.[168] Empirical studies are not conformant of whether intellectual property favor innovation and economic growth or not, but yet the intellectual property framework builds upon said assumption by being a deviation of competition.[169]

Preserving a right to reverse engineer can also be underpinned by the incentive to do follow-on innovations, in which innovation is built and enhanced based on previous innovations through the use of reverse engineering.[170] The argument can be related to the interest of technology distribution which underlie other intellectual property rights such as patents wherein exclusivity is given in exchange of having the innovation publicly disclosed.[171] Technology distribution can namely be said to be facilitated by reverse engineering since the practice enables other practitioners to obtain and learn from existing innovations to the benefit of society.[172]

---

[164] See Eriksson, I. O., & Hettne, J. (2011). *EU-rättslig metod Teori och genomslag i svensk rättstillämpning.* Stockholm: Norstedt Juridik AB, p. 168.
[165] See section 3.4 in this thesis.
[166] Samuelsson, P. Scotchmer, S. (2002). *The Law and Economics of Reverse Engineering.* The Yale Law Journal, volume 111:1575, 1575–1663, p. 1615.
[167] Dreier, T. (1991). *The Council Directive of 14 May 1991 on the Legal Protection of Computer Programs.* European Intellectual Property Review. 13(9) 319-330, p. 6.
[168] See Palmer, A. K., Vinje, T. C. (1992). *The EC Directive on the legal protection of computer software: New law governing software development.* Duke Journal of comparative & international law, volume 2:65, 65-87, p. 69-71.
[169] Compare different opinions expressed in Levin. M. (2017). *Lärobok i immaterialrätt.* Stockholm: Wolters Kluwer Sverige AB, p. 35, Davis, R., Kapor, M. D., Reichman, J. H., & Samuelsson. (1994). *A Manifesto concerning the Legal Protection of Computer Programs.* Columbia Law Review, Vol. 94:2308. 2308-2431, p. 2366, and Kungl. IngenjörsVetenskapsAkademien (IVA). (2011). *Immateriella Rättigheter och Ekonomiska Incitament En kortfattad översikt av kunskapsläget.* Stockholm: Kungl. Ingenjörsvetenskapsakademien (IVA), p. 6.
[170] Davis, R., Kapor, M. D., Reichman, J. H., & Samuelsson. (1994). *A Manifesto concerning the Legal Protection of Computer Programs.* Columbia Law Review, Vol. 94:2308. 2308-2431, p. 2393.
[171] See e.g. Swedish Patent Act (1967:837).
[172] See Davis, R., Kapor, M. D., Reichman, J. H., & Samuelsson. (1994). *A Manifesto concerning the Legal Protection of Computer Programs.* Columbia Law Review, Vol. 94:2308. 2308-2431, p. 2393.

4.3.4 Conclusion

I believe the appropriate interpretation lies in addressing the permissibility of contractual restrictions of reverse engineering and decompilation in relation to the SD. Such an interpretation enables contractual provisions to comply with both directives, which is not the case when relying on the TSD, since the TSD enables a free right to restrict reverse engineering which makes article 5(3) and 6 of the SD irrelevant. I also believe relying on the TSD, and thus promoting an unlimited right to restrict reverse engineering, will most likely cause more harm than good in regards of competition and distribution of knowledge and technology in the software industry because of mentioned importance of interoperability.

As addressed in the background of this thesis, it is apparent that relying on the SD in relation to reverse engineering and decompilation, entails a possibility (although minor) to restrict the practices. One could thus say that by relying on the SD in relation to reverse engineering and decompilation, one balances the interests of society concerning technology distribution and competition, against the interests of private actors wanting to recoup on their R&D investments.[173] I believe so, since the rights of reverse engineering and decompilation are preserved, yet possible to restrict to some extent.

Consequently, my answer to RQ(b) is that one should rely on the SD in regards of the permissibility of contractual restrictions of reverse engineering and decompilation.

4.4 RQ(c) Restriction of reverse engineering – To what extent should it be possible?

From RQ(a) we know that reverse engineering and decompilation of know-how as trade secrets in software actualize both the SD and TSD. From RQ(b) we know that one should rely on the SD in regards of the permissibility of contractual restrictions of reverse engineering and decompilation.

This section will thus address to what extent one should be possible to contractually restrict reverse engineering and decompilation of trade secrets in software in relation to both directives, while SD has priority over the TSD in regards of permissibility of contractual restrictions of reverse engineering and decompilation. More explicitly, I will address the possibility to restrict acquisition and usage of trade secrets according to the TSD, while being compliant with the compulsory rights of reverse engineering and decompilation in the SD, see article 5(3), 6 and 8(2) of the SD.

---

[173] See Kungl. IngenjörsVetenskapsAkademien (IVA). (2011). *Immateriella Rättigheter och Ekonomiska Incitament En kortfattad översikt av kunskapsläget.* Stockholm: Kungl. Ingenjörsvetenskapsakademien (IVA), p. 16.

4.4.1 Reverse Engineering

4.4.1.1 Restriction of Acquisition – to what extent?

We know that the right to reverse engineer in article 5(3) of the SD is a right to "observe, study or test the functioning of the program in order to determine the ideas and principles which underlie any element of the program […]" The right to reverse engineer can therefore be compared to a right to *acquire* "ideas and principles which underlie any element of the program […]"[174]

From a copyright perspective we know that the right to reverse engineer in article 5(3) of the SD may only be exercised if two requirements are fulfilled, namely (i) that the user has a right to use a copy of the software, and (ii) that reverse engineering is exercised whilst doing "any of the acts of loading, displaying, running, transmitting or storing the program which he is entitled to do. "[175]

Furthermore, mentioned acts of "loading, displaying, running, transmitting or storing the program […]" may be contractually restricted, provided that the "acts of loading and running [*necessary for the use of a copy of a program*] […]" are not restricted.[176]

Thus, the absolute minimum and compulsory scope of the right to reverse engineer of article 5(3) can be described as a right to "observe, study or test the functioning of the program in order to [*determine/acquire*] the ideas and principles which underlie any element of the program […]" while doing the "acts of loading and running [*necessary for the use of a copy of a program*] […]"[177]

A contractual provision which restricts acts of "loading, displaying, running, transmitting or storing the program […]" but does not restrict "acts of loading and running [*necessary for the use of a copy of a program*] […]" should thus be deemed legitimate since it respects the compulsory scope of the right to reverse engineer according to the SD. If a licensee breaches such a legitimate contractual provision, there should be question of both copyright infringement and unlawful acquisition of trade secrets according to article 4 of the SD, and 3(1)(b) and (d), and 4(2) of the TSD.

---

[174] See article 5(3) of the SD
[175] Ibid.
[176] Case C-406/10, *SAS Institute Inc. v. World Programming Ltd.,* paragraphs 58 and 62, Opinion of Advocate General BOT Delivered on 29 November 2011; Case C-406/10 SAS Institute Inc. v World Programming Ltd., paragraph. 96, and article 5(3) and recital 13 of the SD.
[177] Ibid.

I believe Shemtov comes to a different conclusion, since Shemtov does not address the legitimate restriction of reverse engineering presented by the CJEU in the case of *SAS Institute Inc. v. World Programming Ltd*.[178] I do however agree with Shemtov regarding the illegitimacy of a complete restriction of reverse engineering. A complete contractual restriction of reverse engineering which also restricts the "acts of loading and running [*necessary for the use of a copy of a program*] […]" will namely be difficult to justify both normatively as well as teleologically considering CJEU's interpretation of the purpose of 5(3) of the SD. The CJEU did state that the copyright holder should not extend the protection of ideas and principles by using a license agreement.[179] A complete restriction of reverse engineering by relying on trade secret protection will namely have the same effect of protecting ideas and principles through a license agreement independent of whether the license agreement relies on copyright or trade secret protection. I therefore believe that such a contractual clause should be deemed a circumvention of the right to reverse engineer in article 5(3) of the SD and thus be deemed void and null according to article 8(2) of the SD, just as Shemtov suggests.[180]

Consequently, I draw the conclusion – that as long as the right to reverse engineer in article 5(3) of the SD is not further restricted than what was considered a permissible restriction in *SAS Institute Inc. v. World Programming Ltd*., the restriction should be deemed legitimate.[181]

4.4.1.2 Restriction of Use – to what extent?

Article 5(3) of the SD does not impose anything about a right to *use* the information that has been *determined/acquired* through reverse engineering. Hence, there is no explicit normative nor compulsory ground which prevent restriction of *use* of information determined/acquired through reverse engineering according to article 5(3) and 8(2) of the SD.

Shemtov addressed the possibility to restrict *usage* of information determined through reverse engineering, in which Shemtov says that article 8(1) of the SD which states that the SD shall not affect other regulations including those concerning trade secrets, can enable an interpretation in which a contractual clause can restrict *usage* of information by relying on the TSD instead of the SD. Shemtov is however later claiming that such a clause will likely be deemed ineffective by the CJEU court due to their pro-competition considerations in *SAS Institute Inc. v. World Programming Ltd* case.[182] I do not agree with Shemtov's latter comment, since my interpretation of the CJEU's reasoning in *SAS Institute Inc. v. World Programming Ltd* case is that the CJEU did not want to enable an extension of the

[178] Shemtov, N. (2017). *Beyond the Code Protection of Non-Textual Features of Software.* Oxford: Oxford University Press, p. 230.
[179] Case C-406/10, *SAS Institute Inc. v. World Programming Ltd*., paragraph. 51.
[180] Shemtov, N. (2017). *Beyond the Code Protection of Non-Textual Features of Software.* Oxford: Oxford University Press, p. 230.
[181] See Case C-406/10, *SAS Institute Inc. v. World Programming Ltd.,* paragraph 62.
[182] Shemtov, N. (2017). *Beyond the Code Protection of Non-Textual Features of Software.* Oxford: Oxford University Press, p. 230.

copyright protection through licensing means and that pro-competition considerations were considered in relation to a possible extension of the scope of copyright protection.[183] The primary reason for their interpretation seemed to rely on the preservation of the idea-expression dichotomy and not motivated by competition interests as such according to my interpretation.

So, should it be legitimate to prevent *usage* of ideas and principles protected as trade secrets? One could argue for such an interpretation, but a more reasonable interpretation would be to not allow contractual provisions which prevents *usage* of ideas and principles that underlie *software interfaces* since such a provision would be in breach with the directive's purpose of *enabling development of interoperable software programs*.[184] Article 5(3) is namely motivated by interoperability interests in relation to article 1(2) of the SD, which indicates that a teleological interpretation of article 5(3) might impose that usage should not be restricted to the extent that it negatively affects interoperability of software.[185] The pro-competition considerations should however not hinder a contractual provision which limits the *use* of trade secret protected ideas and principles that *do not* relate to *software interface information*. *Usage* of information such as the machine learning algorithm should thus be possible to limit through a contractual provision relying on trade secret protection.

Consequently, using a trade secret in breach with a contractual provision that restricts usage of information – *not relating to software interface information*, through e.g. a confidentiality clause, should be deemed an unlawful use of a trade secret according to article 4(3)(b)-(c) of the TSD. My conclusion is therefore that contractual restrictions of *usage* which does not prejudice the interoperability interests of article 5(3) of the SD concerning software interface information, should be deemed normatively motivated.

4.4.1.3 Overall Conclusion

My answer to RQ(c) is thus that one should be possible to contractually restrict reverse engineering accordingly:

> Contractual restriction of the "acts of loading, displaying, running, transmitting or storing the program […]" should be deemed permissible, provided that the "acts of loading and running [*necessary for the use of a copy of a program*] […]" are not restricted.[186] A contractual provision which restricts the *use* of the trade secret protected know-how, e.g. the machine-learning

[183] Case C-406/10, *SAS Institute Inc. v. World Programming Ltd.,* paragraph. 40.
[184] See Dreier, T. (1991). *The Council Directive of 14 May 1991 on the Legal Protection of Computer Programs.* European Intellectual Property Review. 13(9) 319-330, p. 8, on restriction of decompilation.
[185] See Opinion of Advocate General BOT Delivered on 29 November 2011; Case C-406/10 SAS Institute Inc. v World Programming Ltd., paragraph. 92, regarding article 5(3) of the SD's connection with article 1(2) of the SD.
[186] Case C-406/10, *SAS Institute Inc. v. World Programming Ltd.,* paragraphs 58 and 62, Opinion of Advocate General BOT Delivered on 29 November 2011; Case C-406/10 SAS Institute Inc. v World Programming Ltd., paragraph. 96, and article 5(3) and recital 13 of the SD.

algorithm, which *does not* relate to *software interface information* should also be deemed legitimate.

The conclusion can also be said to be in line with the incentives of the software industry, in which a minor possibility to restrict *acquisition* of trade secrets satisfies the interest of technology distribution since follow-on innovation will be possible.[187] Moreover, on a societal level there will likely be a reduction of costs relating to research that try to achieve the same thing without learning from already existing innovations.[188] On the other hand, the fact that there is a possibility to restrict reverse engineering at all, can be said to accredit the interest of turnover since the software owner will be possible to have some control over trade secrets in software, which might promote transactions of the software in license relationships.[189]

The possibility to restrict *usage* of information which do not relate to software interface information can also be said to balance the interests of turnover, innovation and competition in a legitimate way. I believe so, since the mere possibility to restrict usage, enables licensing models such as the one Med-Company Z might engage in.[190] It might also enable a prolonged leading time, which all in all can be said to promote Med-Company Z to perform additional innovation. However, the fact that the possibility to restrict usage is restricted, acknowledges the interest of interoperability for promoting competition.[191]

4.4.1.4 Med-Company Z's Possibility to Restrict Reverse Engineering

If we reconnect to the case of this thesis, we know that black-box reverse engineering does not require anything but using the software.[192] It should therefore be possible to perform black-box reverse engineering while performing the "acts of loading and running [*necessary for the use of a copy of a program*] […]" in article 5(3) of the SD. Hence, there are difficulties for Med-Company Z to contractually prevent a licensee from *determining/acquiring* the machine learning algorithm using black-box reverse engineering. However, considering my foregoing analysis, it should be possible to restrict the *usage* of the machine learning algorithm once it has been acquired/determined, through e.g. a confidentiality clause, since the machine learning algorithm does not relate to software interface

[187] See Davis, R., Kapor, M. D., Reichman, J. H., & Samuelsson. (1994). *A Manifesto concerning the Legal Protection of Computer Programs.* Columbia Law Review, Vol. 94:2308. 2308-2431, p. 2393.
[188] See Kungl. IngenjörsVetenskapsAkademien (IVA). (2011). *Immateriella Rättigheter och Ekonomiska Incitament En kortfattad översikt av kunskapsläget.* Stockholm: Kungl. Ingenjörsvetenskapsakademien (IVA), p. 6.
[189] See Kungl. IngenjörsVetenskapsAkademien (IVA). (2011). *Immateriella Rättigheter och Ekonomiska Incitament En kortfattad översikt av kunskapsläget.* Stockholm: Kungl. Ingenjörsvetenskapsakademien (IVA), p. 6, and Adlercreutz, A., Gorton, L., & Lindell-Frantz, E. (2016). *Avtalsrätt 1.* Lund: Juristförlaget, p. 19.
[190] Ibid.
[191] See Samuelsson, P. Scotchmer, S. (2002). *The Law and Economics of Reverse Engineering.* The Yale Law Journal, volume 111:1575, 1575–1663, p. 1615.
[192] Shemtov, N. (2017). *Beyond the Code Protection of Non-Textual Features of Software*. Oxford: Oxford University Press, p. 71.

information. Consequently, the extent to which Med-Company Z should be possible to practically restrict a relevant method of reverse engineering, such as black-box reverse engineering, is to restrict usage of information which does not relate to software interface information.

The following section will address to what extent contractual restriction of decompilation should be normatively motivated in relation the SD when the TSD complies with the SD.

4.4.2 Decompilation

4.4.2.1 Restriction of Acquisition – to what extent?

We know that decompilation is a right to perform the acts of "reproduction of the code and translation of its form within the meaning of points (a) and (b) of Article 4(1) […]" without any authorization from the right holder – provided that mentioned acts are indispensable to obtain software interfaces, see recital 10 in comparison with article 6 (1) of the SD.[193] The right to decompile can thus be said to be a right to *acquire* software interface information.

Furthermore, the right to *obtain/acquire* software interface information may only be exercised if the acts under article 6 of the SD are indispensable for obtaining/acquiring mentioned information and if the requirements in 6(1) (a)-(c) of the SD are fulfilled.[194]

The compulsory scope of decompilation in article 6(1) of the SD is hence much more explicit compared to the right to reverse engineer since it may be derived from the article itself. The following should thus not be possible to contractually restrict according to article 6(1) and 8(2) of the SD.

> The indispensable acts of "reproduction of the code and translation of its form within the meaning of points (a) and (b) of Article 4(1) […]" for obtaining/acquiring software interface information in relation to the requirements in 6(1) (a)-(c).

One could however argue that the prerequisite of *indispensability* in article 6(1) of the SD, enables an interpretation according to which it should be possible to contractually restrict *non-indispensable* acts

[193] Influenced by opinions expressed in Shemtov, N. (2017). *Beyond the Code Protection of Non-Textual Features of Software*. Oxford: Oxford University Press, p. 85, and Dreier, T. (1991). *The Council Directive of 14 May 1991 on the Legal Protection of Computer Programs.* European Intellectual Property Review. 13(9) 319-330, p. 7.
[194] Dreier, T. (1991). *The Council Directive of 14 May 1991 on the Legal Protection of Computer Programs.* European Intellectual Property Review. 13(9) 319-330, p. 7.

of decompilation.[195] Consequently, if Med-Company Z would disclose the software interface information of their software, decompilation would no longer be indispensable for obtaining/acquiring the software interface. And according to article 6 of the SD, a user would then need authorization from the copyright holder (Med-Company Z) in order to perform the acts of decompilation.

Conclusively, if a contractual clause would restrict the compulsory scope of article 6 of the SD, it would both be in contrary to the normative formulation, and the purpose of the article – to *enable acquisition of software interface information in order to promote interoperability of software.*[196] The provision would thus most likely be deemed void and null according to article 8(2) and 6 of the SD. It should therefore not be possible to claim unlawful acquisition of trade secrets according to 3(1)(b) and 4(2) of the TSD based on such a provision.[197]

My conclusion is thus that it should be possible to contractually restrict acquisition of trade secrets by restricting *non-indispensable acts* of decompilation. If a user would decompile in breach with such a provision, there should be a question of both copyright infringement and unlawful acquisition of trade secrets according to article 4 of the SD, and 3(1)(b) and (d) and 4(2) of the TSD.

4.4.2.2 Restriction of Use – to what extent?

We know that there is no compulsory right to use the information obtained through decompilation according to article 6 of the SD.[198] Dreier has furthermore concluded in his comment on the SD, that *software interface information* that has been obtained in compliance with the decompilation right in 6 of the SD cannot be contractually restricted in regards of *usage* by relying trade secret protection.[199] I agree with Dreier's interpretation about restriction of usage of software interface information should not be permitted, since restricting usage of software interfaces would most likely be deemed to contradict the interoperability purpose of the directive and article, which Dreier also underpins in his reasoning.[200]

If we then flip the angle, one should thus be able to argue that a contractual restriction of the right to *use* information obtained through decompilation which *does not* relate to *software interfaces* is justified since such a provision would respect the purpose of interoperability. Such an interpretation would also be normatively supported since article 6 does not include a right to use the information

[195] See Shemtov, N. (2017). *Beyond the Code Protection of Non-Textual Features of Software*. Oxford: Oxford University Press, p. 85, and Dreier, T. (1991). *The Council Directive of 14 May 1991 on the Legal Protection of Computer Programs.* European Intellectual Property Review. 13(9) 319-330, p. 7.
[196] See article 6 of the SD.
[197] See Shemtov, N. (2017). *Beyond the Code Protection of Non-Textual Features of Software.* Oxford: Oxford University Press, p. 230.
[198] Ibid, p. 98.
[199] Dreier, T. (1991). *The Council Directive of 14 May 1991 on the Legal Protection of Computer Programs.* European Intellectual Property Review. 13(9) 319-330, p. 8
[200] Ibid.

obtained/acquired.[201] Consequently, *using* acquired information in breach with a contractual clause such as the one mentioned in the foregoing sentence, should enable a claim of unlawful *use* of a trade secret according to article 4(3)(b)-(c) of the TSD.[202] Thus, my conclusion is that it should be possible to contractually restrict *usage* of information obtained through decompilation by relying on trade secret protection, provided that the information does not relate to *software interfaces.*

4.4.2.3 Overall Conclusion

My answer to RQ(c) is that the extent to which one, e.g. Med-Company Z, should be possible contractually restrict decompilation is the following:

> Contractual restriction of *non-indispensable acts* of decompilation, and/or *usage* of information which does not relate to *software interfaces.*

Similar to my reasoning regarding reverse engineering, I believe this interpretation can be motivated by the incentives and interests of the software industry. I have come to that conclusion since the possibility to prevent decompilation depends on whether the acts of decompilation is indispensable, which imply that software interface information must be possible to obtain elsewhere if the software holder should be entitled to restrict decompilation. I thus think the interpretation acknowledges the incentive of innovation – to keep in control and reap benefits of the work, and the interest of interoperability in favor of competition, without neglecting the other.[203]

Furthermore, the possibility to restrict usage is similar to reverse engineering acknowledging the interest of turnover, since licensing models of software with the possibility to control abstract information is being allowed, without negatively affecting interoperability.[204]

4.4.2.4 Med-Company Z's Possibility to Restrict Decompilation

What does this mean for Med-Company Z? The technical process of decompilation entails the acts of copying/reproduction of object code, as well as translating the object code into source code.[205] The acts correspond to the acts of "reproduction of the code and translation of its form within the meaning of points (a) and (b) of Article 4(1) […]" which cannot be restricted by the right holder (Med-Company Z) if they are indispensable for acquiring software interface information, see article 6 of the SD.

---

[201] Shemtov, N. (2017). *Beyond the Code Protection of Non-Textual Features of Software.* Oxford: Oxford University Press, p. 98.
[202] Compare Ibid, p. 230 in which Shemtov addresses the possibility to restrict usage of information obtained through decompilation.
[203] See Kungl. IngenjörsVetenskapsAkademien (IVA). (2011). *Immateriella Rättigheter och Ekonomiska Incitament En kortfattad översikt av kunskapsläget.* Stockholm: Kungl. Ingenjörsvetenskapsakademien (IVA), p. 6, and, Samuelsson, P. Scotchmer, S. (2002). *The Law and Economics of Reverse Engineering.* The Yale Law Journal, volume 111:1575, 1575–1663, p. 1615.
[204] Ibid.
[205] Litman, J. (1992). *Copyright and Information Policy.* Law and Contemporary Problems, Vol. 55: No 2. 185-209, p. 198.

Med-Company Z should therefore not be possible to restrict the indispensable acts for acquiring software interface information. However, Med-Company Z has the choice of disclosing software interface information, thus omitting the right to decompile.

In regards of usage of information obtained through decompilation, it should be possible to restrict usage of information which does not relate to software interface information through e.g. a confidentiality clause.

Consequently, Med-Company Z should be possible to entirely prevent decompilation by disclosing software interface information – and preferably enlighten the licensee about the disclosure and state that decompilation is no longer indispensable for obtaining software interface information and therefore prohibited. Furthermore, Med-Company Z should be possible to contractually restrict use of information obtained which do not relate to software interface information, by using e.g. a confidentiality clause.

4.4.3 Suggested Contractual Provisions

This section aims to suggest contractual provisions based on what I suggest to be the possibility to contractually restrict reverse engineering and decompilation. The suggested provisions will be less explicit than the analysis in order to have the licensee bearing the burden of acting within its right to reverse engineer and decompile.

4.4.3.1 Restriction of Acquisition

The following suggestion regulates reverse engineering and decompilation in regards of acquisition of trade secrets:

> You may not perform reverse engineering or decompilation except as and only to the extent foregoing restriction is prohibited by applicable law.[206]

Mentioned provision respects the licensee's right reverse engineer – right to "observe, study or test the functioning of the program in order to determine the ideas and principles which underlie any element of the program […]" while doing the "acts of loading and running [*necessary for the use of a copy of a program*] […]", since the provision includes a disclaimer which states that applicable law may limit the provision.[207] The provision does also respect the right to decompile – *the right to perform indispensable acts* of "reproduction of the code and translation of its form within the meaning of points (a) and (b) of

[206] Influenced by Apple's provision referred to in Shemtov, N. (2017). *Beyond the Code Protection of Non-Textual Features of Software.* Oxford: Oxford University Press, p. 12, in relation to ITunes EULA http:// www.apple.com/ legal/ itunes/ appstore/ dev/ stdeula/
[207] Article 5(3) of the SD, Case C-406/10, *SAS Institute Inc. v. World Programming Ltd.,* paragraphs. 54-55 and 58, and recital 13 of the SD.

Article 4(1) […]" in relation to the requirements in 6(1)(a)-(c) for acquiring software interface information, due to mentioned disclaimer.

If one wants to prevent decompilation further, one can include a sentence which states that software interface information has been published and that decompilation therefore is prohibited by the licensor.

4.4.3.2 Restriction of Use

When it comes to usage of information, it is possible to regulate the matter with a non-disclosure agreement such as the WIPO template referred to in the background of this thesis.[208] The non-disclosure agreement should define what is considered confidential information, and thus include algorithms in the definition to cover the machine learning algorithm of Med-Company Z's software. The reason for doing so is to take reasonable steps for preserving secrecy, see requirement for trade secret protection in article 2(1) of the TSD. The agreement should further stipulate how the information may be used, for what purpose, and for what period the information is classified as confidential etc.[209]

Consequently, information obtained from reverse engineering or decompilation can be regulated as follows:

> Confidential Information that has been obtained from reverse engineering, decompilation or any other use of the software may only be used in compliance with attached Non-Disclosure Agreement, except as and only to the extent foregoing restriction is prohibited by applicable law.[210]

Mentioned provision respects the right of reverse engineering and decompilation by not restricting the use of information that underlie software interfaces, since the provision includes a disclaimer which states that applicable law may limit the provision.

4.5 Closing Words

We have now addressed three legal areas in relation to the matters of software and reverse engineering, including decompilation, and have also come to conclusions regarding to what extent one should be possible to contractually restrict reverse engineering and decompilation of trade secrets in software. The

---

[208] WIPO. (2019). *Disclosing Confidential Information*. Downloaded 2019-04-24 from https://www.wipo.int/sme/en/documents/disclosing_inf_fulltext.html#P53_4509

[209] Ibid.

[210] Influenced by Apple's provision referred to in Shemtov, N. (2017). *Beyond the Code Protection of Non-Textual Features of Software.* Oxford: Oxford University Press, p. 12, in relation to ITunes EULA http:// www.apple.com/ legal/ itunes/ appstore/ dev/ stdeula/

suggested way of solving the problem statement of this thesis sheds light on the fact that the solution must not be of absolute character, i.e. either completely for or against contractual restrictions of reverse engineering in software. Instead, I have suggested a solution which enables companies such as Med-Company Z to restrict some aspects of reverse engineering and decompilation, whereas some aspects cannot be restricted. The suggested interpretation enables Med-Company Z to restrict usage of information obtained from black-box reverse engineering and decompilation as long as the information does not relate to software interface information, and the practice of decompilation can be completely prohibited if software interface information has been disclosed. Conclusively, Med-Company Z would be able to engage in future license relationships such as the opportunity presented to the company in the practical example, since Med-Company Z would be able to control the use of the machine learning algorithm as a trade secret in a license relationship. The possibility to remain in control does further imply that Med-Company Z could extract financial capital from mentioned relationships without risking losing the algorithm, and thus continue with performing innovation and business development relating to the algorithm in the future.

5 Sources

5.1 Articles

Augustin, M., Fritz, M., Oh, S. J., & Schiele, B. (2018). *Towards Reverse-Engineering Black-Box Neural Networks.* ICLR 2018 Conference paper.

Band, J. (2018). *The Global API Copyright Conflict.* Harvard Journal of Law & Technology, volume 31 Special Issue Spring 2018, 616-637.

Boyle, J. (2003). *The second enclosure movement and the construction of the public domain.* Law and Contemporary Problems, 33-74.

Clark, B., Martinis, L. D., & Niebel, R. (2018). *The EU Trade Secrets Directive: all change for trade secret protection in Europe?* Journal of Intellectual Property Law & Practice, vol. No. 6, 445-457.

Davis, R., Kapor, M. D., Reichman, J. H., & Samuelsson. (1994). *A Manifesto concerning the Legal Protection of Computer Programs.* Columbia Law Review, Vol. 94:2308. 2308-2431.

Dreier, T. (1991). *The Council Directive of 14 May 1991 on the Legal Protection of Computer Programs.* European Intellectual Property Review. 13(9) 319-330.

Johnson-Laird, A. (1994). *Software Reverse Engineering in the Real World.* Dayton. L. Rev. (19) 843-902.

Juels, A., Reiter, M. K., Ristenpart, T., Tramér, F., & Zhang, F. (2016). *Stealing Machine Learning Models via Prediction APIs.* Available through USENIX, Open Access Media.

Kungl. IngenjörsVetenskapsAkademien (IVA). (2011). *Immateriella Rättigheter och Ekonomiska Incitament – En kortfattad översikt av kunskapsläget.* Stockholm: Kungl. Ingenjörsvetenskapsakademien (IVA).

Litman, J. (1992). *Copyright and Information Policy.* Law and Contemporary Problems, Vol. 55: No 2. 185-209.

Palmer, A. K., Vinje, T. C. (1992). *The EC Directive on the legal protection of computer software: New law governing software development.* Duke Journal of comparative & international law, volume 2:65, 65-87.

Samuelsson, P. Scotchmer, S. (2002). *The Law and Economics of Reverse Engineering.* The Yale Law Journal, volume 111:1575, 1575–1663.

Schröder, V. (2018). *Transformation from a Patchwork Quilt to a Unified Fabric: Discussion on a Few Particular Aspects of the New Finnish Trade Secrets Act and the EU Trade Secrets Directive.* Nordiskt Immateriellt Rättsskydd, NIR 4 / 2018, 500-515.

5.2 Books

Adlercreutz, A., Gorton, L., & Lindell-Frantz, E. (2016). *Avtalsrätt 1.* Lund: Juristförlaget.

Bernitz, U., Pehrson, L., Rosén, J., & Sandgren, C. (2017). *Immaterialrätt och otillbörlig konkurrens*. Stockholm: Jure Förlag.

Butterfield, A., & Ngondi, G. E. (2016). *A dictionary of Computer Science.* Oxford: Oxford University Press. Software.

Eriksson, I. O., & Hettne, J. (2011). *EU-rättslig metod Teori och genomslag i svensk rättstillämpning.* Stockholm: Norstedt Juridik AB.

Fahlbeck, R. (2019). *Lagen om skydd för företagshemlighet En kommentar och rättsöversikter.* Stockholm: Norstedt Juridik AB.

Kernighan, B. E., & Ritchie, D. M. (1988). *The C Programming Language.* New Jersey: Precentice Hall.

Levin. M. (2017). *Lärobok i immaterialrätt.* Stockholm: Wolters Kluwer Sverige AB.

Maunsbach, U. Wennersten, U. (2018). *Grundläggande immaterialrätt.* Malmö: Gleerups Utbildning AB.

Kleineman, J. (2018). Rättsdogmatisk metod, (pp. 21-46). Nääv, M., & Zamboni, M. (Red.), *Juridisk metodlära*. Lund: Studentlitteratur AB.

Petrusson, U. (2004). *Intellectual Property & Entrepreneurship Creating Wealth in an Intellectual Value Chain.* Gothenburg: CIP.

Sandgren, C. (2018). *Rättsvetenskap för uppsatsförfattare ämne, material, metod och argumentation.* Stockholm: Norstedts Juridik AB.

Shemtov, N. (2017). *Beyond the Code Protection of Non-Textual Features of Software.* Oxford: Oxford University Press.

Wolk, S. (2016). *Datorprogramalster i upphovsrätten*. Uppsala: Iustus Förlag AB.

5.3 Court cases

Case C-5/08, (2009). *Infopaq International A/S v. Danske Dagblades Forening.*

Case C-406/10, (2012). *SAS Institute Inc. v. World Programming Ltd*.

5.4 Legislation and International Treaties

Council Directive 91/250/EEC of 14 May 1991 on the legal protection of computer programs.

Directive 2001/29/EC of the European Parliament and of the Council of 22 May 2001 on the harmonisation of certain aspects of copyright and related rights in the information society.

Directive 2009/24/EC of the European Parliament and of the Council of 23 April 2009 on the legal protection of computer programs (Codified version).

EU Directive 2016/943 of the European Parliament and of the Council of 8 June 2016 on the protection of undisclosed know-how and business information (trade secrets) against their unlawful acquisition, use and disclosure.

Swedish Patent Act (1967:837).

Swedish Trade Secret Act (2018:558).

The Agreement on Trade-Related Aspects of Intellectual Property Rights, Annex 1C of the Marrakesh Agreement Establishing the World Trade Organization, signed in Marrakesh, Morocco, on 15 April 1994.

5.5 Preparatory Works and Communication from the EU

European Commission Staff Working Document Analysis of measures that could lead significant market players in the ICT sector to license interoperability information. SWD(2013) 209 final.

European Commission Staff Working Document Impact Assessment *Accompanying the document* proposal for a Directive of the European Parliament and of the Council on the protection of undisclosed know-how and business information (trade secrets) against their unlawful acquisition, use and disclosure. SWD(2013) 471 final.

European Parliament Council Commission, Interinstitutional Agreement of 22 December 1998 on common guidelines for the quality of drafting of Community legislation, General Principles 10.

Official Journal of the European Communities L 122 Volume 34. 17 May 1991.

Opinion of Advocate General BOT Delivered on 29 November 2011; Case C-406/10 SAS Institute Inc. v World Programming Ltd.

5.6 Preparatory works from Sweden

Prop 2017/18:200. En ny lag om företagshemligheter.

SOU 2017:45. En ny lag om företagshemligheter.

5.7 Websites

Berkeley Economics University of California. (2015). *Suzanne Scotchmer.* Downloaded 2019-04-24 from https://www.econ.berkeley.edu/profile/suzanne-scotchmer

Berkeley Law University of California. (2019). *Pamela Samuelsson.* Downloaded 2019-04-24 from https://www.law.berkeley.edu/our-faculty/faculty-profiles/pamela-samuelson/

Johnson-Laird Inc. (2017). *About Andy.* Downloaded 2019-04-29 from https://www.jli.com/about.html

Oxford University Press. (2019*). Beyond the Code Protection of Non-Textual Features of Software.* Downloaded 2019-04-23 from https://global.oup.com/academic/product/beyond-the-code-9780198716792?cc=se&lang=en&#

PWC. (2019). *Why AI and robotics will define New Health.* Downloaded 2019-04-23 from https://www.pwc.com/gx/en/industries/healthcare/publications/ai-robotics-new-health.html

Schröder, V. (2017, January). *Decompiling the rules on trade secrets, software and reverse engineering*. Downloaded 2019-02-03, from https://trustinip.com/decompiling-the-rules-on-trade-secrets (website no longer available).

W3Schools. (2019). *HTML DOM write() Method.* Downloaded 2019-04-24 from https://www.w3schools.com/jsref/met_doc_write.asp

WIPO. (2019). *Disclosing Confidential Information*. Downloaded 2019-04-24 from https://www.wipo.int/sme/en/documents/disclosing_inf_fulltext.html#P53_4509